

# Halstead Complexity

YEGOR BUGAYENKO

Lecture #4 out of 24

80 minutes

The slidedeck was presented by the author in this [YouTube Video](#)

All visual and text materials presented in this slidedeck are either originally made by the author or taken from public Internet sources, such as web sites. Copyright belongs to their respected authors.



“Any attempt to find a universal set of metrics that could be applied to any computer program, might at first glance appear destined to be unfruitful, if not merely difficult. But, without universal, measurable parameters, we would be in the position of trying to develop the science of thermodynamics before the advent of a temperature scale.”

— Maurice H. Halstead. *Elements of Software Science (Operating and Programming Systems Series)*. Elsevier Science Inc., 1977. doi:[10.5555/540137](https://doi.org/10.5555/540137)

## Inputs

- $\eta_1$  — the number of distinct operators
- $\eta_2$  — the number of distinct operands
- $N_1$  — the total number of operators
- $N_2$  — the total number of operands

## Example from Wikipedia

### Example [\[edit\]](#)

Consider the following C program:

```
main()
{
    int a, b, c, avg;
    scanf("%d %d %d", &a, &b, &c);
    avg = (a+b+c)/3;
    printf("avg = %d", avg);
}
```

The distinct operators ( $\eta_1$ ) are: `main`, `()`, `{}`, `int`, `scanf`, `&`, `=`, `+`, `/`, `printf`, `,`, `;`

The distinct operands ( $\eta_2$ ) are: `a`, `b`, `c`, `avg`, `"%d %d %d"`, `3`, `"avg = %d"`

## Operators and Operands

“When a program is translated from one language to another, as from FORTRAN to machine language for example, the actual operators and operands may indeed change, but both versions must still consist of combinations of operators and operands. No other category of entities need be present.”

Source: Maurice H. Halstead. Advances in Software Science. *Advances in Computers*, 18(1):119–172, 1979.  
doi:[10.1016/S0065-2458\(08\)60583-5](https://doi.org/10.1016/S0065-2458(08)60583-5)

## Length and Vocabulary

- $N_1 + N_2 = \text{Length}$
- $\eta_1 + \eta_2 = \text{Vocabulary}$
- $\eta_1 \times \log_2 \eta_1 + \eta_2 \times \log_2 \eta_2 = \text{Estimated Length}$

## Length vs. Vocabulary



“The size of a program, regardless of the metric used to measure size, is a function of the vocabulary of the program.”

Source: Charles P. Smith. A Software Science Analysis of Programming Size. In *Proceedings of the Annual Conference*, pages 179–185, 1980.

[doi:10.1145/800176.809965](https://doi.org/10.1145/800176.809965)

## Volume, Difficulty, and Effort

- $N$  = Length
- $\eta$  = Vocabulary
- $N \times \log_2 \eta$  = Volume
- $\eta_1/2 + N_2/\eta_2$  = Difficulty
- $D \times V$  = Effort



## Effort vs. Understandability

“We have independently tested the hypothesis that the mental effort required to create a program (measured by  $E$ ) is related to a person’s ability to understand a program or to find bugs in existing programs. The studies of Gould and Weissman as well as our work strongly support these hypotheses.”

Source: Ann Fitzsimmons and Tom Love. A Review and Evaluation of Software Science. *ACM Computing Surveys (CSUR)*, 10(1):3–18, 1978. doi:[10.1145/356715.356717](https://doi.org/10.1145/356715.356717)

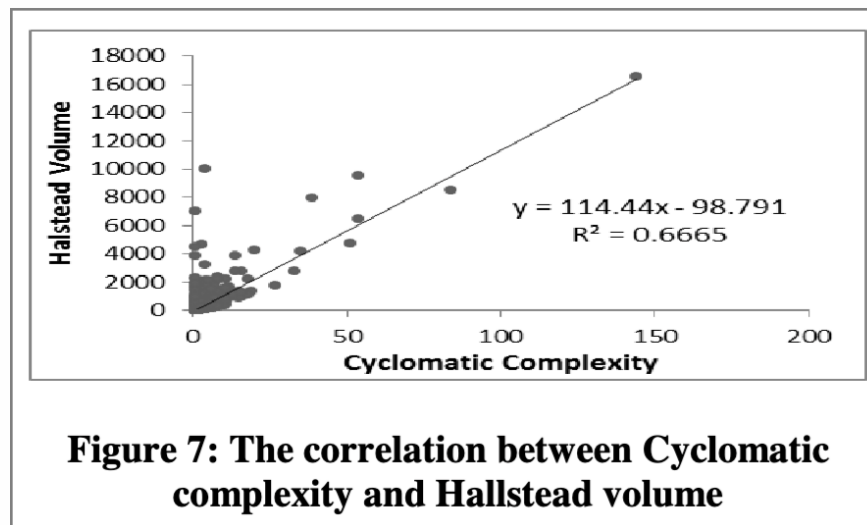


BILL CURTIS

“In the current study, the complexity metrics were more highly related to the performance of less experienced programmers. Thus, the complexity metrics may not represent the most important constructs for predicting the performance of experienced programmers. These programmers probably conceptualized programs at a level other than that of operators, operands, and basic control paths.”

— Bill Curtis, Sylvia B. Sheppard, Phil Milliman, M. A. Borst, and Tom Love. Measuring the Psychological Complexity of Software Maintenance Tasks With the Halstead and McCabe Metrics. *IEEE Transactions on Software Engineering*, 5 (2):96–104, 1979. doi:[10.1109/TSE.1979.234165](https://doi.org/10.1109/TSE.1979.234165)

## CC vs. Halstead Volume

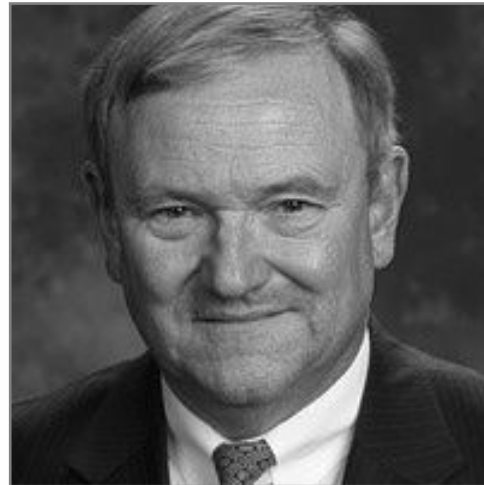


“From the Fig. 7, it is obvious that that the correlation between Cyclomatic Complexity and Halstead Volume is strong. The change in Cyclomatic Complexity will impact on Halstead Volume and vice versa.”

Source: Yahya Tashtoush, Mohammed Al-Maolegi, and Bassam Arkok. The Correlation Among Software Complexity Metrics With Case Study, 2014

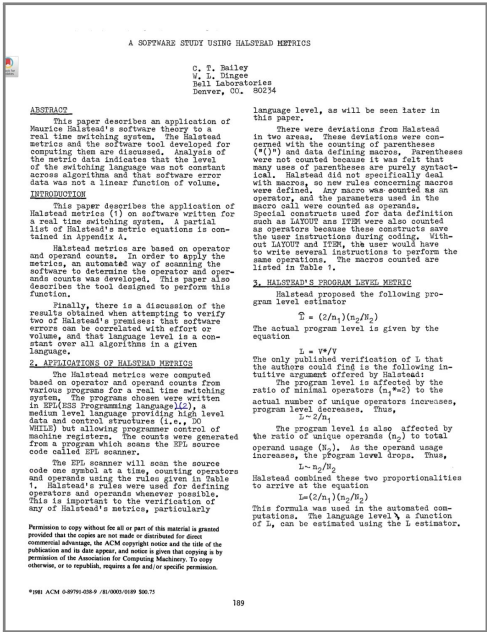
## Time and Bugs Estimate

- $E$  = Effort
- $E/18$  = Time (in seconds)
- $V/3000$  = Bugs



“In studying some error data provided us by Rome Air Development Center, Phil Milliman and I found Halstead’s metric a remarkably accurate predictor of delivered bugs in a system developed with modern programming practices and tools.”

— Bill Curtis. Program Complexity and Software Errors: A Front End for Reliability, 1979



“The authors analysis indicated that software errors were not a linear function of program volume. It is felt that errors are a result of many other factors besides those which make up the volume metric. Perhaps the correlation exists on a very large scale (over an entire software system).”

— C. T. Bailey and W. L. Dingee. A Software Study Using Halstead Metrics. In *Proceedings of the Workshop/Symposium on Measurement and Evaluation of Software Quality*, pages 189–197, 1981. doi:[10.1145/800003.807928](https://doi.org/10.1145/800003.807928)

Halstead Complexity is supported by a few tools:

- multimetric for C++, Java, Python, and many others
- JHawk (not free) for Java
- Halstead Metrics Tool for Java
- PhpStorm for PHP

# Bibliography

C. T. Bailey and W. L. Dingee. A Software Study Using Halstead Metrics. In *Proceedings of the Workshop/Symposium on Measurement and Evaluation of Software Quality*, pages 189–197, 1981. doi:[10.1145/800003.807928](https://doi.org/10.1145/800003.807928).

Bill Curtis. Program Complexity and Software Errors: A Front End for Reliability, 1979.

Bill Curtis, Sylvia B. Sheppard, Phil Milliman, M. A. Borst, and Tom Love. Measuring the Psychological Complexity of Software Maintenance Tasks With the Halstead and McCabe Metrics. *IEEE Transactions on Software Engineering*, 5(2):96–104, 1979. doi:[10.1109/TSE.1979.234165](https://doi.org/10.1109/TSE.1979.234165).

Ann Fitzsimmons and Tom Love. A Review and Evaluation of Software Science. *ACM Computing Surveys (CSUR)*, 10(1):3–18, 1978. doi:[10.1145/356715.356717](https://doi.org/10.1145/356715.356717).

Maurice H. Halstead. *Elements of Software Science (Operating and Programming Systems Series)*. Elsevier

Science Inc., 1977. doi:[10.5555/540137](https://doi.org/10.5555/540137).

Maurice H. Halstead. Advances in Software Science. *Advances in Computers*, 18(1):119–172, 1979. doi:[10.1016/S0065-2458\(08\)60583-5](https://doi.org/10.1016/S0065-2458(08)60583-5).

Charles P. Smith. A Software Science Analysis of Programming Size. In *Proceedings of the Annual Conference*, pages 179–185, 1980. doi:[10.1145/800176.809965](https://doi.org/10.1145/800176.809965).

Yahya Tashtoush, Mohammed Al-Maolegi, and Bassam Arkok. The Correlation Among Software Complexity Metrics With Case Study, 2014.