An experiment comparing Fortran programming times with the software physics hypothesis

by R. D. GORDON and M. H. HALSTEAD
Purdue University
Lafayette, Indiana

ABSTRACT
Recent discoveries in the area of Algorithm Structure or Software Physics\textsuperscript{1-25} have produced a number of hypotheses. One of these relates the number of elementary mental discriminations required to implement an algorithm to measurable properties of that algorithm, and the results of one set of experiments confirming this relationship have been published.\textsuperscript{16} That publication, while significant, made no claim to finality, suggesting instead that further experiments were warranted. This paper will present the results of a second set of experiments, having the advantages of being conducted in a single implementation language, Fortran, from problem specifications readily available in computer textbooks.

The first section of this paper presents the timing hypothesis, and the elementary equations upon which it rests. The second section presents the details of the experiment and the results which were obtained, and the third section contains an analysis of the data.

TIMING HYPOTHESIS

Measurable properties of any implementation of any algorithm include:

\begin{align}
\eta_1 &= \text{The count of distinct operators} \\
\eta_2 &= \text{The count of distinct operands} \\
N_1 &= \text{Total uses of operators} \\
N_2 &= \text{Total uses of operands}
\end{align}

The vocabulary, \( \eta \), is given by:

\( \eta = \eta_1 + \eta_2 \)

and the length, \( N \), is:

\( N = N_1 + N_2 \)

From these properties, it is possible to obtain the volume, \( V \), in bits, as:

\( V = N \log_2 \eta \)

and the implementation level, \( L \), where \( L \leq 1 \), as:

\( L = \frac{\eta^*}{\eta_1 \eta_2} \)

where \( \eta^* \), the minimum possible number of operators, will equal 2 for most algorithms. (One for the name of a function, plus one for a grouping symbol operator). It has been shown\textsuperscript{1} that the product \( L \times V \) is invariant under translation from one language to another, and that for programs without impurities:\textsuperscript{3,6,8}

\( N = \eta_1 \log_2 \eta_1 + \eta_2 \log_2 \eta_2 \) (5)

From this point, the following nine steps yield the timing equation:

1. A program consists of \( N \) selections from \( \eta \) elements.
2. A binary search of \( \eta \) elements requires \( \log_2 \eta \) comparisons.
3. A program is generated by making \( N \log_2 \eta \) comparisons.
4. Therefore, the volume, \( V \), is a count of the number of comparisons required.
5. The number of elementary mental discriminations required to complete one comparison measures the difficulty of the task.
6. The level, \( L \), is the reciprocal of the difficulty.
7. Therefore, \( E \), the count of elementary mental discriminations required to generate a program, is given by:

\( E = \frac{V}{L} \) (6)

8. \( S \), the speed with which the brain makes elementary mental discriminations can be obtained from psychology\textsuperscript{25} as:

\( 5 \leq S \leq 20 \) discriminations per second.

9. Therefore, the time to generate a preconceived program, by a concentrating programmer, fluent in a language, is:

\( \hat{T} = \frac{V}{SL} \) (7)

Equation 7 may be expressed in more basic terms by substituting for \( V \) from equation 3, and for \( L \) from equation 4, with \( \eta^* = 2 \), giving:

\( \hat{T} = \frac{\eta N_1 N \log_2 \eta}{2 S \eta_2} \) (8)
The effect of possible impurities\(^5\) may be eliminated from equation 8 by substituting for \(N\) from equation 5. Letting \(S = 60 \times 18 = 1080\) will then give, for time in minutes:

\[
T = \frac{\eta_1 N_1 (\eta_2 \log_{10} N_2 + \eta_2 \log_{10} N_2) \log_{10} \eta_2}{2160 \eta_2} \tag{9}
\]

Each of the variables on the right hand side of equation 9 can be readily measured (or counted) in any computer program, and the experiment described in the next section was designed to compare results from that equation with observed programming times.

### EXPERIMENTAL PROCEDURE

Eleven problems were arbitrarily selected from two published sources. In selecting candidates for the experiment, problems were sought which were stated in a non-procedural form. Further, the problem statement had to be complete. That is, in the course of solving a particular problem, specific laws of physics, mathematics, etc. would not have to be derived. The problems finally selected were taken from Knuth,\(^27\) and from Maurer and Williams,\(^28\) and cover a wide range of topics including character manipulation, list processing, simulation experiments and mathematical analysis. The source of each problem statement is cited in Table I.

On each of eleven days, one of these problems was implemented by the senior author. In order to maintain a consistent level of performance all work was conducted in a quiet room, free from distractions, during the same period of the day. The time required to fully implement the problem was obtained. This total time included the number of minutes spent reading the statement of the problem, preparing flowcharts and writing preliminary versions of the code, writing the final version of the code, desk checking, and the time spent working to correct errors in the program. Time to keypunch was not included.

![](https://www.computerhistory.org)

For a number of reasons, including availability and fluency, all of the algorithms were implemented in Fortran. In the course of solving a problem the correctness of the implementation was checked by executing a sufficiently complex test case for which a correct answer was known. In some cases the solution to a problem was written as a subroutine and testing required that a main routine be written. In such a case only the preparation of the subroutine was considered for the experiment. In addition, several implementations made use of subroutines previously written. Such routines were also not included.*

After each program was completed, a careful count was made to determine values of \(\eta_1, \eta_2, N_1, \text{ and } N_2\). In obtaining these values all read, write, declarative statements and comments were ignored. The results are shown in Table I.

### ANALYSIS OF THE DATA

The programming time predicted by theory was obtained for each program by applying equation 9 to the data in Table I. This result, \(T\), can be compared with the observed value, \(T\), in Table II. In addition, a count of the number of statements in each program was obtained, and the programs were ordered according to these values.

The average of the calculated values, 34 minutes, is fortuitously close to the observed value, 35 minutes. The coefficient of correlation is 0.934, only slightly smaller than the value of 0.952 reported in an earlier experiment.\(^16\) In further agreement with that experiment, the correlation between length and observed times, 0.887, is lower than between observed and calculated times.

In conclusion, it may again be observed that one more set of experimental data does not contradict the simple hypothesis. As a result, further carefully controlled experiments by others would appear to be warranted.

* Additional details available from the author.

### TABLE I—Experimental Data

<table>
<thead>
<tr>
<th>Program Specifications</th>
<th>Software Parameters</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>No.</td>
<td>Ref.*</td>
<td>Page</td>
</tr>
<tr>
<td>G1</td>
<td>K</td>
<td>158</td>
</tr>
<tr>
<td>G2</td>
<td>K</td>
<td>159</td>
</tr>
<tr>
<td>G3</td>
<td>K</td>
<td>196</td>
</tr>
<tr>
<td>G4</td>
<td>K</td>
<td>377</td>
</tr>
<tr>
<td>G5</td>
<td>K</td>
<td>158</td>
</tr>
<tr>
<td>G6</td>
<td>K</td>
<td>154</td>
</tr>
<tr>
<td>G7</td>
<td>M</td>
<td>32</td>
</tr>
<tr>
<td>G8</td>
<td>M</td>
<td>32</td>
</tr>
<tr>
<td>G9</td>
<td>M</td>
<td>88</td>
</tr>
<tr>
<td>G10</td>
<td>M</td>
<td>89</td>
</tr>
<tr>
<td>G11</td>
<td>M</td>
<td>27</td>
</tr>
</tbody>
</table>

* \(K = \text{Knuth,}^27\), \(M = \text{Maurer and Williams.}^28\)

### TABLE II—Experimental Results

<table>
<thead>
<tr>
<th>Program Number</th>
<th>Statement Count</th>
<th>Programming Time-Minutes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(T) observed</td>
<td>(T) Equ. 9</td>
</tr>
<tr>
<td>G7</td>
<td>7</td>
<td>5</td>
</tr>
<tr>
<td>G11</td>
<td>8</td>
<td>5</td>
</tr>
<tr>
<td>G6</td>
<td>11</td>
<td>21</td>
</tr>
<tr>
<td>G5</td>
<td>15</td>
<td>30</td>
</tr>
<tr>
<td>G3</td>
<td>18</td>
<td>16</td>
</tr>
<tr>
<td>G1</td>
<td>18</td>
<td>19</td>
</tr>
<tr>
<td>G10</td>
<td>59</td>
<td>91</td>
</tr>
<tr>
<td>Means</td>
<td>35.0</td>
<td>34.1</td>
</tr>
</tbody>
</table>

From the collection of the Computer History Museum (www.computerhistory.org)
REFERENCES


Additional References:
