USING STUDENT SURVEYS IN DETERMINING THE DIFFICULTY OF PROGRAMMING ASSIGNMENTS*

Andres Alvarez
Colorado State University and Televent

Terry A. Scott
University of Northern Colorado

ABSTRACT

This paper deals with the subject of determining difficulty in programming assignments for introductory Computer Science courses. A pilot study was performed in an introductory Computer Science course as the University of Northern Colorado. In this study students were given surveys asking them to predict upcoming assignment difficulty and also to grade past assignment difficulty. These surveys were used in conjunction with several code metrics as well as the students' grades. While the sample size of students was small, some interesting findings were observed. Primarily, students are decent at predicting difficulty in assignment as a group. Second, the biggest impact on students' perception of difficulty is the length of their solution. Finally, the biggest impact on students' performance on an assignment is the number of flow of control constructs in their solution.

INTRODUCTION:

One difficulty in generating computer programming assignments for students is determining the difficulty of each assignment. If the assignment is too easy, the students aren't challenged and they don't learn enough. If too hard, many students don't complete the assignment and there is less learning. If the instructor does the computer program it can help in deciding how difficult the assignment is, but this is certainly not completely reliable and depends greatly on students' preparation and understanding. When discussing the difficulty of a coding assignment we should make clear that there are two kinds of coding difficulty measures. There is some absolute amount of difficulty that has to do with how hard it is to actually code the algorithm for expert programmers. There also is a relative difficulty that has to do with the preparation of the students/programmers. For

---

the classroom we really are interested in the second kind of difficulty determination. Our interest is to see the difficulty of the assignments through the students' perspective.

A key motivator in this study is using real classroom experience as a way of attaining our data. As such, data from the Fall Session of 2009 of "Introduction to Computer Science" at the University of Northern Colorado was used. This data is ideal as it is from a classroom setting using direct feedback from the students on difficulty. One problem the researchers can identify is the small sample size in the class. Only eighteen students completed the course and many students did not complete every survey or assignment. However, given the positive results of the study, we are confident in moving forward with a similar experiment on a larger sample size in the near future.

The rest of this paper is organized as follows. First, the experimental design and key variables are reviewed. Next, some findings from our pilot study are highlighted. Finally, some remarks about our initial study and thoughts for going forward with a larger sample size are discussed.

EXPERIMENT AND VARIABLES:

Our experiment involves two key aspects. The first was giving students surveys to provide feedback on assignment difficulty. The second was gathering all relevant data from the assignment solutions. This relevant data included the students' surveys, the source code and the students' grades on the assignments. This process involved the following steps where the steps in bold represent steps that were added and were not normal in previous sessions of the class.

1. Assignment was handed out in class and reviewed.
2. **With the assignment the student was given a survey asking them to rate on a scale of 1 to 5 how difficult they think the assignment will be.**
3. The student was given 8 days to complete the assignment.
4. The student handed assignment in. Assignments were due on Saturdays.
5. Student's work was graded and by Sunday their grade and class distribution on the assignment was available via the class website.
6. **In Monday's class, after the grade results of the assignment were available, students filled out a survey asking them to rate how difficult the assignment was on a scale of 1 to 5. Additionally they were asked to order all completed assignments in order of easiest to hardest.**

The researchers obtained IRB approval for collecting the surveys and source code from the students. Students were provided with a waiver to elect not to have their data be used in this experiment. No students opted out of the experiment. Students filled out surveys using an alias. The teacher was unaware of the mapping of the alias to the student's name during the semester. This was to done to prevent teacher bias in grading of assignments. The surveys were handed out for all class assignments. However, only the data and surveys from programming assignments were used in this study. A total of ten assignments were given but only six directly involved coding.

For the sake of completeness the other assignments used CircuitMaker[1] software for the first two labs[1]. This allowed students to create virtual digital circuits such as

adders to illustrate how a computer does its work at the electronic level. For the third assignment the students did assembly code on a virtual computer[2]. For the last lab students programmed a virtual Turing machine[3]. You may look at all assignments by visiting the web-pages for the class[4].

The following data were available to the researchers for addressing the problem of assignment difficulty:

- The students' source code.
- The students' grades.
- The students' surveys.

To evaluate the students' surveys in a quantitative manner the researcher's used the following metrics:

- The pre-assignment (Pre) difficulty rating of the assignment.
- The post-assignment (Post) difficulty rating of the assignment.
- The rank of the assignment (Rank) relative to the other programming assignments.

To evaluate the students' source code in a quantitative manner the researchers employed the following metrics:

- The total number of lines of code (LOC). This excludes comments and empty line
- The number of flow-of-control (FOC) constructs. These are loops and decision statements.
- The number of function definitions. (Functions)
- The number of variables. (Variables)
- The grade received on each assignment with 10 points being the maximum with a possible 1 or 2 points of extra credit added to that. (Grades)

The above metrics combined with the surveys and grades were used to determine what factors make assignments hard both in terms of performance via grades and perception via surveys. While the above variables are by no means exhaustive, they do supply a large amount of data per student and do allow for much analysis of introductory programming assignments.

RESULTS:

Lab #	Pre	Post	LOC	FOC	Functions	Variables	Grades
4	2.8	2.8	15.2	0.2	2.9	4.1	11.2
5	3.0	2.9	17.8	0.6	0.5	7.4	8.8
6	3.5	3.8	35.9	1.3	6.6	11.6	10.5
7	4.2	3.9	44.8	3.9	4.1	35.1	7.8

8	4.0	4.3	41.9	3.8	5.6	10.7	7.7
9	4.0	4.2	48.9	5.2	4.0	14.5	7/4

Table 1: Student Averages by Assignment

	Lab #	**Pre**	**Post**	**LOC**	**FOC**	**Functions**	**Variables**	**Grades**
Lab #	1	0.89	0.93	0.93	0.96	0.79	0.41	-0.84
Pre	0.89	1	0.93	0.97	0.93	0.57	0.74	-0.79
Post	0.93	0.93	1	0.96	0.88	0.70	0.46	-0.68
LOC	0.93	0.97	0.96	1	0.93	0.72	0.64	-0.72
FOC	0.96	0.93	0.88	0.93	1	0.73	0.57	-0.86
Functions	0.79	0.57	0.70	0.72	0.73	1	0.11	-0.41
Variables	0.41	0.74	.046	..64	0.57	0.11	1	-0.52
Grades	-0.84	-0.79	-0.68	-0.72	-0.86	-0.41	-0.52	1

Table 2: Correlation between Variables

First Question: What drives survey results?

Note that the diagonal contains all 1's indicating the correlation between the variable and itself is always one (no surprise). The first aspect of the post-assignment survey results as a whole is the fact that they seem moderately predictive. Four variables have a correlation of greater than 0.89; the Lab #, Pre, LOC and FOC. The Lab # is indicative that as the semester progresses the assignments get harder. The pre-assignment survey results are promising in that it may be possible that students can predict assignment difficulty. In the case of the Pre, there are 5 variables that seem to have significant correlation.

Two of our five coding metrics that had the highest correlation, were LOC and FOC. LOC indicates a longer program and it would appear this variable has the single largest factor in influencing students' view of the assignment difficulty. We also note that the FOC also correlates well with student difficulty ratings. This is a good finding as this indicates more complicated programs as measured by the number of flow-of-control statements and also has a good correlation with student performance.

Second Question: What drives grades?

The first important thing to point out is that grades are not as predictable as how difficult a student will think an assignment is. It is important to notice that all correlation values are negative when related to grades. This means an increase in any of the variables has a negative impact on the students' grades. With that in mind we do notice two variables that have a correlation of less than -0.8. The first is the Lab #. This follows conventional thought nicely. As the class progresses the assignments get harder. As the assignments get harder the students will not perform as well. We notice that the code metric that has the biggest influence on grade is the "Flow of Control", FOC, statements. These statements increase the complexity of code. This is certainly interesting to consider. As an assignment's complexity increases the average grade decreases.

A final interesting note is that students' pre-assignment survey results correlate better than post assignment survey results. The grade with post-assignment survey results was actually surprising to the researchers. Students had full access to their grade and the class distribution before filling out their post-assignment survey. It may be that this did not influence their view of the assignment. It is also possible that not all students viewed their grade before filling out the survey.

Third Question: How reliable are these results?

Student	Variable	Metric	Correlation	Full Samples	Total Possible Samples
2	FOC	Post	0.85	5	6
2	LOC	Post	0.89	5	6
10	FOC	Post	0.83	5	6
10	LOC	Post	0.93	5	6
17	FOC	Post	0.95	5	6
17	LOC	Post	0.89	5	6

Table 3: Correlation between Student Post Survey Results and FOC or LOC Variables

While using averages for all of the variables certainly produced some interesting results, the authors should point out the issue of the sample size used. Only eighteen students completed the course and some of them missed various parts of the study. For instance many of the students missed at least one survey and several students failed to turn in one or more assignments.

When looking at individual students the trends we noticed on the average across all students are not there for individual students. We looked at each student that completed at least five sets of post-assignment surveys and assignments. We then looked at which

students had a correlation of greater than 0.8 for any of the variables compared to their post-assignment difficulty ratings. We also looked for students with a correlation of less than -0.8 for any of the variables compared to their grades. We only found three students that met these criteria for assignment difficulty rating and no student met these criteria for their grades.

One suggestion for why there is variability for individual students is that for each student there are many factors not controlled. Such things could include the student's health during the semester or conflicts with other classes. These factors could influence whether a student performs well on an assignment. It is very believable that these factors could average out over an entire class, but would be hard to account for in an individual student. In this study it is important to note that an individual student from assignment to assignment is not as reliable as a group of students. This can give some sway to the argument of using large studies as we are proposing instead of using case studies on a small number of students.

FUTURE PLANS:

Given the findings of this experiment, the researchers are looking to continue it with a few changes. First, we will be doing the survey experiment on a different course that has more programming assignments. Additionally this course has two sections, each with roughly thirty students. This will occur in the fall semester of 2010 and will provide a much larger sample size. Second, we will use an Internet web site for collecting survey results. The web site will be time sensitive relative to the assignments' hand-out dates and their due dates. These two changes should provide a much larger data set and hopefully help alleviate some of the missing surveys we observed in this study.

CONCLUSION:

This experiment was done as a pilot to figure out what we needed to do when we did the experiment with larger number of students. However a few things that we learned and expect to be confirmed when we have a larger number of students are:

- Number of lines of code (LOC) in a program gives at least a good estimate of the difficulty of the assignment as perceived by the students. The author did a study earlier on program clarity and complexity and found LOC was a reasonably good estimator of program complexity and much easier to compute than some complexity metrics[5].
- In this case at least the number of flow-of-control statements also is an important factor in considering student performance.
- As the semester continues the assignments became more difficult as perceived by the students. This is what the instructor sought to do.
- Students are able to make a good estimate of the difficulty of assignments at least as judged by LOC and pre-assignment results.
- The students' averages were predictable. Individual student results were much harder to predict.

As a professor what can you take away from this study. The authors believe several things if in fact these results are born out with a larger sample size (also note these may only apply to an introductory programming course):

1. Students can generally tell if an assignment will be difficulty even before they begin the assignment.
2. The number of lines of code and/or the number of control structures in an assignment are good predictors of the difficulty of assignments in an or at least the students view of difficulty of an assignment.
3. The results for individual students were not well correlated with the results as determined by correlations on the class as a whole. This certainly flies in the face of case studies that are many times done in qualitative research.

BIBLIOGRAPHY

[1] http://www.aboutspice.com/details-219, MicroCode Engineering, January, 1998.

[2] http://users.dickinson.edu/~braught/kands/kands.html, Department of Mathematics and Computer Science, Dickinson College, 5/14/2008.

[3] Scott, Terry A., "Turing Machine Simulation Used in a Breadth First Computer Science Course", *Journal of Computing Sciences in Colleges*, Vol 22, October 2006.

[4] http://cs.unco.edu/course/CS101/F09/Schedule101.html, Department of Mathematical Sciences, University of Northern Colorado, September, 2009.

[5] Scott, Terry A, "Size and Complexity Metrics and Introductory Programming Students", *The Journal of Computing in Small Colleges*. Vol 11, No 2, 1995, pp 165-173.

[6] http://www.r-project.org, The R statistical package was used for computing all correlations.