Investigating Factors of Student Learning in Introductory Courses

Matthew Hertz
Computer Science Department
Canisius College
Buffalo, NY 14208
hertzm@canisius.edu

Sarah Michele Ford
Department of Sociology
University of Massachusetts, Amherst
Amherst, MA 01003
ford@soc.umass.edu

ABSTRACT

Instructors of the introductory computer science courses, commonly called "CS1" and "CS2", face a large number of choices when designing their classes. Instructors have available to them a multitude of ways to explain each topic as well as course-wide choices such as objects-first or objects-late or using a functional or procedural language. Understanding how these options can affect student learning would help simplify these decisions. Unfortunately, just comparing how well students perform may not be accurate as it ignores the many confounding factors that could also have made a difference. To get beyond that problem, this study investigates underlying factors that affect student learning.

Using a survey of instructors, we find that students' abilities are nearly always correlated with the importance that the instructor placed on a particular topic. Our results also highlight several "hard" topics for which student mastery and topic importance were not correlated in CS1 and only weakly correlated in CS2. While one might expect the time spent covering a topic in class to also be correlated with student mastery, we find little evidence of this. In fact, for some basic programming concepts, we document *negative* correlations between instructional time and learning. We discuss how instructors can use these results when organizing their courses and how the computer science education community can use this finding of "hard" topics to focus their efforts.

Categories and Subject Descriptors

K.3.2 [**Computers and Education**]: Computers and Information Science Education—*Computer science education; Curriculum*

General Terms

Measurement, Standardization

Keywords

CS1, CS2, Survey, Curriculum Design

1. INTRODUCTION

When designing a course, CS1 and CS2 instructors must consider the relative importance of and how much class time to devote to each of the topics they cover. These decisions then influence, and are influenced by, a number of other pedagogical options, such as the programming language used or if and when to introduce objects. Following the course, all of these choices may be adjusted or the entire focus of the course changed to try to improve student learning. But knowing which change or changes to make is difficult. Research exists examining each individual option and, generally, shows that switching to or using any option improves student learning, but can they all be "right"? Little has been done to understand why all these choices work and what the underlying factors are that influence student learning.

Finding that the importance an instructor places on a topic, and not the amount of time spent on that topic, is important to student learning and would help instructors make these decisions. Understanding these details will also help identify "difficult" or "hard to teach" topics – topics where learning is not correlated with either time or importance. With this identification, the computer science education community can focus its attention on those areas where it is most needed.

To uncover these relationships, this paper uses a survey that was taken by 58 CS1 instructors and 41 CS2 instructors teaching classes that represent the full range of class sizes, languages, paradigms and teaching pedagogies. The survey asked instructors about their course practices including the class time spent and importance placed on a range of topics. This survey also asked instructors how well their students mastered each topic. Using these data, we examine what relationship, if any, exists between the time spent or importance placed on a topic and the resulting level of student ability. The level of correlation will be tested in two different ways. Spearman's ρ determines if any monotonic correlation exists between the data; Pearson's r determines if there exists a linear correlation. Pearson's r also allows us to compute r^2, the percentage of the variation in students' abilities that can be explained by differences in the level of importance or class time spent on a topic.

For nearly all of the topics we examined, we will show that students' abilities at the completion of the course are closely correlated with the level of importance the instructor placed on that topic. This result suggests that students usually follow their instructor's lead and focus on those topics the instructor believes to be most important. Two commonly taught lessons in CS1, SUBROUTINES/FUNCTIONS and TYPES, do not exhibit this relationship, however, suggesting that they are very difficult to teach. Unlike importance and learning, both CS1 and CS2 had few instances where the time spent teaching a topic and student learning were closely correlated. While usually not statistically significant, we did find a number of topics which

Control constructs	Methods, functions, subroutines
Variables, types, expressions	Arrays, strings
I/O (e.g., files, printing to screen)	Style (e.g., indentation, variable names)
Object-oriented programming (e.g., inheritance, class anatomy)	Abstraction mechnisms (e.g., interfaces, abstract classes)
Testing	Debugging
Error handling (e.g., exceptions)	Recursion
Sorting	Stacks, queues, lists
Trees	Graphs
Other data structures	

Table 1: Listing of topics used in the survey. These topics were derived from CS1 and CS2 topics detailed in the LACS Model Curriculum.

even exhibited negative correlations. These correlations suggest that one could improve student learning by *reducing* the time spent teaching these topics.

These results highlight factors that help explain the results found in computer science education research. While the amount of time spent on a topic may not make of a difference, the importance placed on a topic clearly does. By changing a class to focus on different topics, instructors should find that student learning on those topics improves. This analysis also provides quantitative data documenting topics that are "hard". With this in mind, future researchers could document which factors enable students to master these topics and disseminate these best practices.

We begin by presenting related works in Section 2. The design and methodology of the survey and an analysis of our respondents are in Section 3. The results of our survey appear in Section 4. We analyze these results and discuss future directions for this research in Section 5. Our conclusions are presented in Section 6.

2. RELATED WORK

Several papers have presented the range of approaches that are used to teach introductory programming courses. Model curricula, such as those developed by the ACM-IEEE committee [13] and the Liberal Arts Computer Science Consortium [8], not only guide curricular choices, but also map out how the curricula should be applied to different paradigms (e.g., objects-early, objects-late, functional, scripting-based). Other work has used surveys to investigate the different programming languages used in CS1 and CS2 [4], the time spent covering a list of topics [2], level of emphasis instructors place upon topics [5] and level of student learning on a variety of topics [6]. All of these past works document the current practices of instructors. Unlike this work, however, they do not attempt to examine how these factors affect student learning nor do they use this to determine what topics are most difficult to teach or learn.

Other past work has investigated which of the introductory computer science concepts are the most difficult. The SIGCSE Bulletin contained two articles by Dale [2, 3] in which she investigated which topics instructors felt were the most difficult. Goldman, et al., provided another approach to determine which topics were most difficult [5]. This work used a survey-driven multi-step Delphi process to enable a panel of 20 experts to reach consensus on the level of difficulty for a number of topics. Like our current work, these past studies used surveys to find the difficult topics. Unlike the previous studies, however, our research determines topic difficulty by analyzing respondents' instructional practices to find where student learning cannot be correlated with instructors' efforts.

3. SURVEY DESIGN & RESPONSE

We now discuss how we constructed the survey and present an overview of the courses taught by our respondents. Other details about the design of this survey can be found in our previous work [6].

3.1 Survey Design

We began designing the survey by developing a list of topics that would appear in introductory computer science courses. The ACM-IEEE Model Curriculum [13] provided an obvious starting point, but the curriculum only contains high-level topics and would limit possible analyses. We therefore began with the more detailed topic listing from the LACS Model Curriculum [8]. While the LACS Model Curriculum is focused on programs offering liberal arts degrees, it is based upon the ACM-IEEE Model Curriculum and so we felt its concept listing would apply broadly. Our previous work documents the process we used to determine the final list of topics [6] and these topics can be found in Table 1.

This research analyzes the current practices of CS1 and CS2 instructors. This required gathering as many responses, from as wide a range of instructors and institutions, as possible. For the questions asking instructors for the amount of time they spent covering each topic, respondents selected from "Not a part of this course/No time", "1 hour", "2 hours", "3 hours", …, "9 hours", "10+ hours", and "Don't Know". For responses to how important the respondent felt the topic was to their course and the level of student mastery at the conclusion of the course we used Likert-type scales [7] and included a "Don't Know" option. For levels of importance, the possible answers were: "Unrelated", "Tangential", "Beneficial", "Important", "Critical", and "Don't Know". For levels of mastery, respondents could choose between "None", "Limited", "Capable", "Knowledgeable", "Mastered", and "Don't Know".[1] When analyzing the data, we pruned responses on a topic when the respondent answered "Don't Know" or did not respond to all questions for that topic. We also pruned responses to topics instructors recorded as being "Unrelated" to their course. Including these data could not help inform us about what affects student learning or identify difficult to teach topics and so could only serve as a potential bias.

3.2 Survey Responses

Requests for instructors to complete the survey were sent periodically to the SIGCSE e-mail discussion list. Using an approach known as targeted snowball sampling [1], each announcement included a link to the survey website and a request that the mail be forwarded to appropriate instructors and related mailing lists. This led to 143 unique survey responses. After removing incomplete and abandoned surveys, we were left with 99 valid responses. 58 of these completed surveys were from CS1 instructors and 41 were from

[1]Purely objective measures of mastery are not feasible. This type of analysis would require language-independent tests of CS1 and CS2 concepts such as that proposed in [11,12]. Unfortunately, these tests are not publicly available and could not be given to a large enough sample to analyze the range of current teaching practices.

instructors of CS2 courses. (Some respondents filled out surveys for both CS1 and CS2. The data were entered separately and we do not analyze data across courses, so we count these respondents as instructors of both CS1 and CS2.)

Language Used	**CS1**	**CS2**
Java	55%	59%
C++ (with objects)/C#	12%	2%
Pascal/C/C++ (without objects)	14%	22%
Lisp	12%	7%
Other	7%	10%

Class Size	**CS1**	**CS2**
Min. - Max.	9 - 150	4 - 150
Mean	37	27
Median	23	20
1st - 3rd Quartile	18 - 39	10 - 30
	CS1	**CS2**

Table 2: The range of teaching approaches used by instructors responding to our survey. Though not directly comparable, the use of languages we find is similar to those found in earlier surveys [4]. Though we know of no census of class sizes, our responses cover a diverse range of courses.

Table 2 shows the diversity of the courses taught by the respondents. Because our survey was not limited to US-based institutions, a direct comparison with a previous survey of educational practices [4] is not possible, but our respondents differ from the results of that earlier survey only in that our respondents are slightly more likely to use either Java or a functional language. The number of students taking respondents' courses also varied from a minimum of 4 to a maximum of 150 students. CS2 courses tended to be much smaller than CS1 courses, but CS2 classes of 75 or more students still made up 10% of our survey.

Our results may be biased by the survey design and participant recruitment. The initial survey solicitations were posted on education-focused discussion lists. While these postings included a request to forward the e-mail, it is likely that most of the instructors who saw this have an active interest in computer science education. It is reasonable to assume the respondents would also place a greater emphasis on teaching than other faculty. While our results may thus skew away from average computer science educators and towards those most focused on student learning, this should only strengthen any conclusions we draw as to which topics are difficult and the effect that classroom time and importance has on student learning.

4. RESULTS

When designing or modifying a course to achieve student learning goals, two factors instructors control are the importance placed on a topic and the amount of class time spent covering each topic. It is reasonable to expect that making a topic more important to a course would increase student learning. Similarly, one would hope and expect that instructional time and levels of student learning are positively correlated. To test for this relationship, we use *Pearson's r* to measure the linear correlation between students' abilities and how important the instructor felt a topic is or how much time was spent on the topic in class. Pearson's *r* provides two different ways to evaluate what correlation, if any, exists. First, the resulting *p* value establishes the statistical significance of the correlation. Additionally, the value of r^2 calculates the percentage of variability in one factor that can be explained by the other [14]. Only linear correlations can be detected by Pearson's *r*, however. Using *Spearman's* ρ we can test for any monotonic relationship [9] and, in combination with the earlier correlation statistic, will highlight cases where supra- or super-linear relationships exist.

4.1 Topic importance & student learning

We first examined the correlation between students' abilities on a topic and the importance the instructor placed on the topic. We found that the correlations were statistically significant for all but a few topics in CS1 courses. Figure 1a illustrates the responses for a typical topic, I/O, which exhibits this correlation. As Figure 1a shows, students did well when the topic was emphasized and had little ability when the topic was considered less important. For the majority of topics in which significant correlations were not found, it was due to the topics being covered by VERY few instructors (84 - 91% of respondents marked these topics as being "unrelated" to their course). Even after including "unrelated" responses, there were 2 topics for which we could not find significant correlations: SUBROUTINES/FUNCTIONS and TYPES. Both topics are commonly taught, all of our respondents covered both of these topics in their courses, but students' abilities were highly variable. As the bubble plot in Figure 1c shows, most instructors think SUBROUTINES/FUNCTIONS are important for their course. This agreement does not translate to equal levels of student learning, however. In fact, only 4% of the variation in students' abilities can be explained by differences in how important instructors felt the topic was. Clearly, teaching SUBROUTINES/FUNCTIONS is *difficult*. But the difficulty of teaching SUBROUTINES/FUNCTIONS pales in comparison to the teaching of TYPES in CS1. As Figure 1d shows, nearly every respondent rated TYPES as either important or very important. Students' abilities are again highly variable, however, with less than 0.5% of the variation in students' abilities being explainable by differences in importance the instructor placed on TYPES.

While the correlation between a topic's importance and student learning is statistically significant for all other topics in CS1, the strength of this correlation varies greatly. For some topics, CS1 students' ability is closely tied to the importance the instructor placed on it. For example, the importance placed on a topic explains nearly 65% of the variation in students' abilities with RECURSION and almost 60% of the variation with ABSTRACTION MECHANISMS. For the plurality of topics in CS1, the correlation is weaker and explains less than half of the variation in student learning.

Unlike in CS1, the responses from CS2 show clear correlations between student learning and topic importance for every topic. But for two of the topics, SUBROUTINES/FUNCTIONS and CONTROL STRUCTURES, this correlation is statistically significant only when using Spearman's ρ, meaning the relationship is monotonic, but not linear. As before, the strength of the correlations varies greatly between topics. A summary of these results for both CS1 and CS2 can be seen in Table 3. More interestingly, the topics with weak correlations found in CS1 carry forward into CS2. In addition, the levels of correlation found in the second course are often lower than those found in first course. That the same topics come up as having the lowest levels of association in both of these courses suggests that these truly are among the most difficult to teach.

4.2 Time spent on a topic & student learning

Students' abilities and the total class time spent on a topic should be closely aligned – one would expect student learning to increase along with the time spent in class working on the topic. Finding low or non-existent levels of correlation would suggest there exist efficient, but not widely used, ways of teaching the topic. This could provide a list of areas ripe for greater scrutiny to find how some instructors do so well or what causes students to struggle with a topic. As the results in Table 4 show, we find that correlations between

		Control Construct.	Procedure/Func.	Types	Arrays	I/O	Style	OO	Abstraction Mech.	Testing	Debugging	Error Handling	Recursion	Sorting	Lists	Trees	Graphs	Other
CS1	*n*	56	54	56	52	53	55	44	31	53	54	39	27	31	19	9	5	15
	ρ	**0.263**	0.197	0.097	**0.389**	**0.609**	**0.314**	**0.660**	**0.753**	**0.719**	**0.638**	**0.526**	**0.809**	**0.575**	**0.666**	0.467	0.356	0.368
	r	**0.307**	0.207	0.062	**0.491**	**0.615**	**0.350**	**0.649**	**0.773**	**0.698**	**0.610**	**0.516**	**0.806**	**0.556**	**0.671**	0.473	0.327	**0.492**
CS2	*n*	38	38	38	38	37	38	37	36	38	38	37	36	35	36	32	23	27
	ρ	**0.522**	**0.349**	**0.574**	**0.578**	**0.594**	**0.552**	**0.577**	**0.675**	**0.416**	**0.505**	**0.546**	**0.373**	**0.578**	**0.336**	**0.457**	**0.752**	**0.693**
	r	0.288	0.174	**0.362**	**0.567**	**0.584**	**0.564**	**0.626**	**0.676**	**0.417**	**0.499**	**0.547**	**0.425**	**0.625**	**0.402**	**0.539**	**0.783**	**0.698**

Table 3: Summary of the correlation of an instructors rating of the importance of a topic and students' abilities at the end of CS1 and CS2 by topic. Values listed in bold are statistically significant ($p \leq 0.5$).

		Control Construct.	Subroutines/Func.	Types	Arrays	I/O	Style	OO	Abstraction Mech.	Testing	Debugging	Error Handling	Recursion	Sorting	Lists	Trees	Graphs	Other
CS1	*n*	55	54	56	52	53	55	44	30	53	54	38	26	31	19	9	5	15
	ρ	-0.143	-0.098	-0.088	0.030	0.239	0.074	**0.385**	**0.779**	**0.433**	**0.266**	**0.619**	**0.659**	0.343	**0.750**	0.506	0.700	**0.790**
	r	-0.146	-0.078	-0.118	0.094	0.151	0.099	**0.367**	**0.699**	**0.330**	0.234	**0.421**	**0.654**	0.340	**0.664**	0.456	0.623	**0.721**
CS2	*n*	38	38	38	38	37	38	37	36	38	38	37	36	35	36	32	23	27
	ρ	-0.193	-0.099	-0.265	**-0.352**	0.053	0.026	-0.030	0.106	0.184	0.178	0.241	-0.105	0.051	-0.177	0.057	**0.466**	**0.487**
	r	-0.172	-0.183	**-0.355**	**-0.335**	0.076	0.038	-0.021	0.163	0.160	0.110	0.224	-0.511	0.102	-0.144	0.145	0.373	**0.547**

Table 4: Summary of the the correlation of the number of hours spent covering a topic and students' abilities at the end of CS1 and CS2 on that topic. Values listed in bold are statistically significant ($p \leq 0.5$).

instructional time on a topic and student learning are ***much*** weaker than correlations between topic importance and student learning levels. For several topics, in fact, we could not document ANY positive correlation between instructional time and student learning. As we now discuss, this pattern can be seen for both CS1 and CS2 courses.

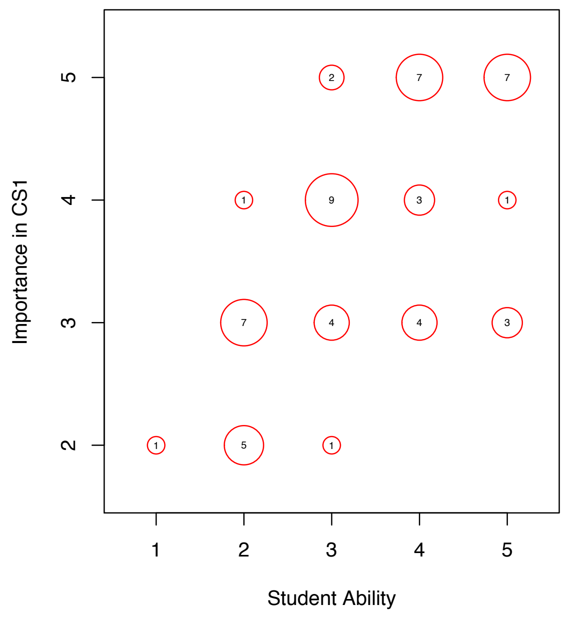
In CS1, there was not a statistically significant correlation between class time spent on a topic and student learning for a majority of topics. Even when the responses labeling a topic "unrelated" are included, we could not find statistically significant evidence of a relationship between teaching time and student learning. This result holds for both the linear and monotonic correlation metrics. Where the correlation is significant, either the strength of this correlation is weak (e.g., TESTING, DEBUGGING) or the topic was taught by a small subset of instructors (e.g., ABSTRACTION MECHANISMS, RECURSION). For three of the most basic topics, CONTROL STRUCTURES, SUBROUTINES/FUNCTIONS, and TYPES, the correlations are weakly negative. While not statistically significant, these correlations suggest that increasing the time spent on these topics is associated with *decreased* student ability. For other commonly taught topics, ARRAYS being the strongest example, the level of correlation means that differences in time spent on a topic could explain as little as 10% of the variations in student ability. In CS1, just spending more time on a subject would result in very little improvement in student learning.

If the correlations seen by CS1 instructors were weakly significant, it is almost impossible for us to conclude that the time spent on a topic in CS2 has any relationship with student learning. In fact, of the four topics for which we find statistically significant correlations, an equal number have negative correlations (TYPES and ARRAYS) as have a positive correlation (GRAPHS and OTHER). Nor are these relationships very strong; only for RECURSION and OTHER does the time spent teaching explain at least 14% of the variation in student ability. For a majority of topics (10 out of 17), in fact,differences in time spent teaching a topic explain less than 3% of the variation in student ability. The two topics in which we see significant positive correlation (GRAPHS and OTHER) are also the only topics which were identified by some respondents as "unrelated" to their course (33% and 20% of respondents, respectively).

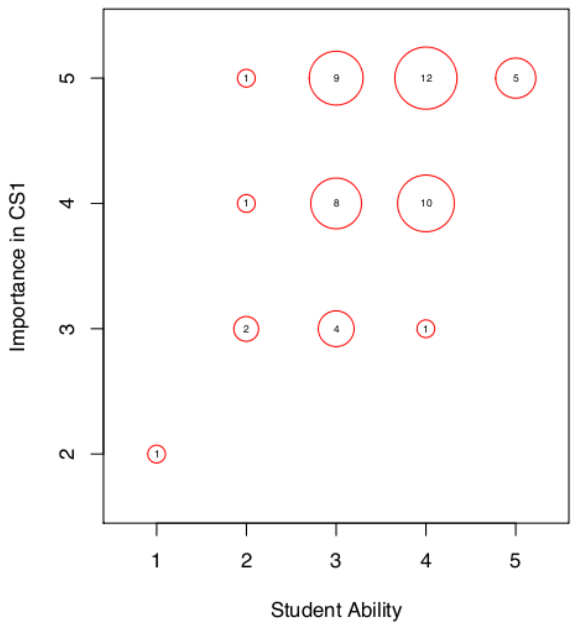
5. ANALYSIS

Survey responses suggest that students' abilities are largely a factor of how important the instructor considered a topic. This result is unsurprising, as the respondents were recruited via a computer science education-related mailing list. This is an important consideration when examining educational studies and assessments. By making a topic more (or less) important, student learning should also be improved (or decreased). While this result is to be expected, it also highlights a fact reflecting the experience of many of our students: some of our most basic concepts are also the ones that are hardest to teach. In CS1, students' abilities on SUBROUTINES/FUNCTIONS and TYPES are unrelated to the importance we place on the subject. That SUBROUTINES/FUNCTIONS is one of the topics in CS2 in which importance and ability are not significantly linearly correlated suggests that something about this topic is truly "hard".

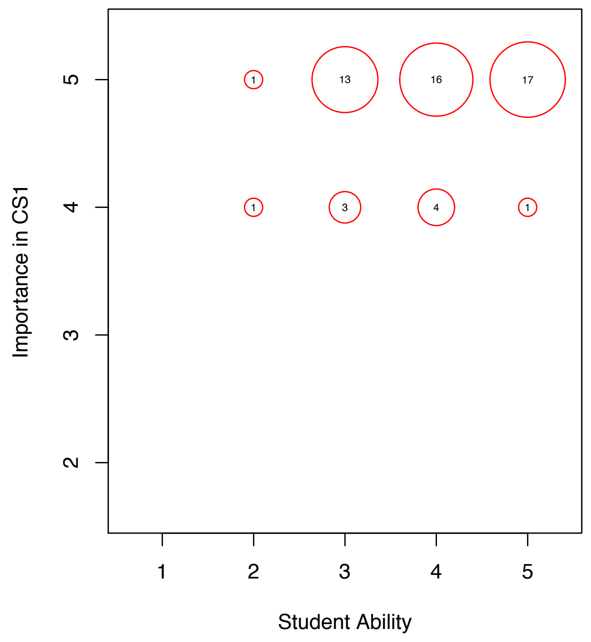
That the results show negative correlations between the time spent on a topic and students' ability ***is*** surprising. That several of the



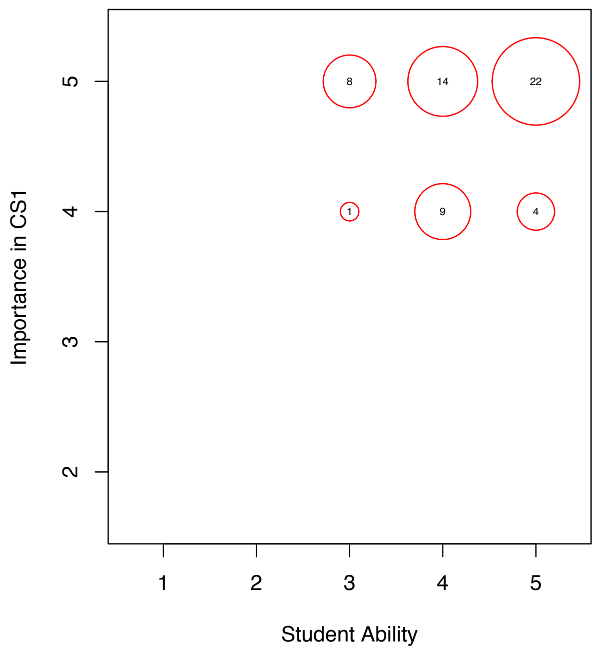
(a) Plot showing the very strong correlation for I/O.



(b) Plot showing a weak, but statically significant, connection for ARRAYS.



(c) Plot showing the weak connection for SUBROUTINES/FUNCTIONS



(d) Plot showing that there is almost no connection for TYPES.

Figure 1: Bubble plots showing a range of relationships between topic importance and student learning in CS1 courses. In each of these graphs, the size of each bubble represents the number of people who responded with that combination of importance and ability. This number is also written at the center of the bubble.

most basic topics have weak, negative correlations in CS1 could reflect a number of possible causes. Differences in how prepared students are at the start of the course, the instructors' comfort with teaching, and the energy and eagerness of the instructor could all help explain these odd results. The lack of significant correlations seen in CS2 may further reflect these differences. That the CS2 correlations are so much weaker than CS1 correlations could also be an indication of differences in students' strength: it is natural to assume that schools with "better" students should need less time to achieve the same level of mastery. These explanations still fail to explain the consistent, albeit weak, negative correlations that appear in CS2. Nor do they explain why the correlations with a topic's importance is so much higher than with the time spent on a topic.

All of this highlights the need for more detailed studies into best teaching practices in introductory courses. Are there approaches or techniques that help students master these topics efficiently? Could topic importance's greater effect on student learning levels be due to student's understanding key ideas or concepts earlier in the course? What underlying ideas or concepts must students possess to learn this material?

Finally, this work emphasizes the need for opportunities such as Ensemble [10] that enable the sharing of teaching material. The results strongly suggest that some instructors have ways of increasing student learning in time efficient ways. In addition to creating a space to share and discuss materials, this work suggests it could help answer important research questions. If posted material specified the time required and resulting level of student mastery, best practices could be identified and disseminated. More importantly, researchers could identify what, if any, common themes exist within those teaching approaches that are the most efficient. This could help improve the teaching of our introductory courses further or create entirely new approaches derived from these best practices.

6. CONCLUSION

Our results show that introductory students' abilities on a topic generally correlate with how important an instructor thought the topic was. This result can help explain the diversity of successful approaches towards teaching these introductory courses. Because the importance placed on each topic changes, our results show that different levels of student learning are to be expected. For nearly every topic, instructors *should* see student learning improve when using an approach that makes the topic more important to the course. We further identified several "hard" topics for which there is no correlation or very weak correlation between importance and ability. These topics – CONTROL STRUCTURES, SUBROUTINES/FUNCTIONS, and TYPES – cover many of the basic programming concepts and were rated as important or very important by nearly all of our respondents. This paper suggests that more research is needed to better understand why these topics are so difficult to teach and investigate better ways to teach these ideas.

Our results also show that there is little correlation between the time spent covering a topic and students' abilities on that topic. In CS1, this is often seen as weak or no correlation for most topics. Worse, in CS2, this is frequently seen as negative correlations suggesting that student learning is improved by decreasing the time topics are taught. There are many possible explanations for this result; future research is needed to understand which explanation or explanations are the cause. Even so, this research highlights the need to share those best practices that best help students learn the material efficiently.

Acknowledgements

The authors would like to thank the survey respondents and anonymous reviewers for their help in improving this paper.

7. REFERENCES

[1] *Implementation and Analysis of Respondent Driven Sampling*, New York, NY, Nov. 2006. Springer New York.

[2] N. B. Dale. Content and emphasis in CS1. *SIGCSE Bulletin*, 37(4):69–73, Dec. 2005.

[3] N. B. Dale. Most difficult topics in CS1: results of an online survey of educators. *SIGCSE Bulletin*, 38(2):49–53, June 2006.

[4] S. Davies, J. A. Polack-Wahl, and K. Anewalt. A snapshot of current practices in teaching the introductory programming sequence. In *SIGCSE '11: Proceedings of the 42nd ACM technical symposium on Computer science education*, pages 625–630, 2011.

[5] K. Goldman, P. Gross, C. Heeren, G. L. Herman, L. Kaczmarczyk, M. C. Loui, and C. Zilles. Setting the scope of concept inventories for introductory computing subjects. *Transactions on Computing Education*, 10(2):1–29, June 2010.

[6] M. Hertz. What do 'CS1' and 'CS2' mean? investigating differences in early courses. In *Proceedings of the 41st SIGCSE technical symposium on Computer science education*, pages 199–203, 2010.

[7] K. Kapp. Likert-type scales: Examples, samples, and information. In *Kapp Notes*, Retrieved 14 Sept. 2010. Available at: `http://www.uleduneering.com/kappnotes/index.php/2010/09/likert-type-scales-examples-samples-and`.

[8] Liberal Arts Computer Science Consortium. A 2007 model curriculum for a liberal arts degree in computer science. *Journal on Educational Resources in Computing*, 7(2):2, 2007.

[9] Spearman's rho is a 'quasi-ordinal' correlation coefficient. Retrieved 26 Aug., 2012. Available at: `https://www.msu.edu/~nurse/classes/summer2002/813/week8spearman.htm`.

[10] F. M. Shipman, L. Cassel, E. Fox, R. Furuta, L. Delcambre, P. Brusilovsky, B. S. Carpenter, G. Hislop, S. Edwards, and D. D. Garcia. Ensemble: a distributed portal for the distributed community of computing education. In *Proceedings of the 14th European conference on Research and advanced technology for digital libraries*, pages 506–509, 2010.

[11] A. E. Tew and M. Guzdial. Developing a validated assessment of fundamental CS1 concepts. In *SIGCSE '10: Proceedings of the 41st ACM technical symposium on Computer science education*, pages 97–101, 2010.

[12] A. E. Tew and M. Guzdial. The FCS1: a language independent assessment of CS1 knowledge. In *SIGCSE '11: Proceedings of the 42nd ACM technical symposium on Computer science education*, pages 111–116, 2011.

[13] The Joint Task Force on Computing Curricula. Computing curricula 2001. *Journal on Educational Resources in Computing*, page 1, 2001.

[14] E. W. Weisstein. Correlation coefficient, Retrieved 26 Aug., 2011. From MathWorld–A Wolfram Web Resource. `http://mathworld.wolfram.com/CorrelationCoefficient.html`.