Teaching CS Unplugged in the High School (with Limited Success)

Yvon Feaster†, Luke Segars‡, Sally K. Wahba†, and Jason O. Hallstrom†
† School of Computing, Clemson University, Clemson SC 29634-0974 USA
‡ Computer Science, UC Berkeley, Berkeley, CA 94720-1776 USA
yfeaste@cs.clemson.edu, lukes@cs.berkeley.edu, {sallyw, jasonoh}@cs.clemson.edu

ABSTRACT

CS Unplugged is a set of active learning activities designed to introduce fundamental computer science principles without the use of computers. The program has gained significant momentum in recent years, with proponents citing deep engagement and enjoyment benefits. With these benefits in mind, we initiated a one-year outreach program involving a local high school, using the CS Unplugged program as the foundation. To our disappointment, the results were at odds with our enthusiasm — significantly. In this paper, we describe our approach to adapting the CS Unplugged material for use at the high school level, present our experiences teaching it, and summarize the results of our evaluation.

Categories and Subject Descriptors

K.3.2 [**Computers and Education**]: Computer and Info. Science Education—*Comp. sci. education, Curriculum*

General Terms

Experimentation, Human Factors

Keywords

CS Unplugged, computer science outreach, high school curriculum, experimental evaluation

1. INTRODUCTION

According to the latest Taulbee Survey, the number of students majoring in undergraduate computer science degree programs rose for the second year in a row [10]. While the news is most welcome, the 14% cumulative increase is relatively modest when considered in context: In 2007, at the base of the enrollment decline in the U.S., computer science enrollment was approximately half of what it was in 2001 [9]. With the U.S. demand for computer scientists expected to increase by 24% over the next decade [3], there is a potential for a shortfall in the domestic labor market. To avoid this shortfall, computer science educators must find new ways to reach out to potential recruits.

One outreach program that is gaining momentum is *CS Unplugged* [4]. The program consists of a set of modules that introduce students to fundamental computer science principles using hands-on activities that are designed to be engaging and fun. The program is distinguished by two characteristics. First, the activities are largely designed to be completed without the use of computers. Instead, students are typically asked to complete kinesthetic tasks that reinforce the concept or algorithmic technique under study (e.g., placing physical objects, moving from one seat to another). Second, the activities are designed to support adaptation to suit a range of learning levels, from grade school students to high school students. Instructors can reveal progressively more detail as appropriate to the target audience.

CS Unplugged offers a compelling set of activities. They break the traditional mold of computer science education and open a new set of possibilities for energizing the K-12 curriculum. The program offers the promise of engagement, excitement, and meaningful learning — and ultimately, a deep supply of potential computer scientists.

With this in mind, we initiated a yearlong outreach program with a local high school based on the CS Unplugged activities. The program consisted of 10 one-hour sessions, repeated for two semesters in an introductory programming course. The evaluation objectives were two-fold. First, we sought to evaluate the impact of the program on student attitudes toward computer science. Second, we sought to evaluate the impact of the program on students' perceived content understanding. We expected the results would point to the obvious conclusion: The program would be a success.

This was not, however, the case. With only one exception, the program had no statistically significant impact on student attitudes or perceived content understanding. Indeed, in some cases, student attitudes moved in an undesirable direction (although these changes were not statistically significant). Given the inclusion of CS Unplugged activities in the ACM K-12 Model Curriculum [1], it is important for experience reports of this kind —both positive *and* negative— to be documented. Educators must not only understand how CS Unplugged can succeed, but also how it can fail.

Paper Organization. Section 2 presents key elements of related work. Section 3 summarizes the structure and content of the outreach program. Section 4 presents the pilot projects. Section 5 presents the results of our evaluation. Finally, Section 6 concludes with a discussion of the evaluation results and potential avenues of further exploration.

2. RELATED WORK

Our goals are shared by a number of other computer science outreach programs. Heersink and Moskal [5] study the attitudes of high school students toward computer science

ITiCSE 2011 Darmstadt, Germany

and information technology. The authors administered a pre-survey to 140 students attending computer science and information technology workshops across 5 states. In relation to the fields of computer science and information technology, the pre-survey measured students' interest level, perceived ability to learn the associated skills, perceived usefulness of the skills, gender issues, and attitudes toward a related professional career. The surveys were written in such a way that the terms "computer science" and "information technology" were interchangeable. Approximately one half of the students received the survey containing "computer science"; the other half received the "information technology" survey. The authors found that students could not distinguish between the two terms. They attributed this confusion to the broad use of educators who serve both as computer science and information technology teachers.

Mano et al. [7] introduce an outreach program at a local middle school, designed to increase student interest in computer science. Volunteers visited 4 classes for approximately 45 minutes each month. Some activities used in this program include disassembling and reassembling a computer, learning how to program with Alice, and using CS Unplugged activities. The authors state that the survey results and volunteers' observations suggest that the program has had a positive impact on student interest in computer science.

Lambert et al. [6] similarly introduce a set of CS Unplugged activities to a group of fourth graders. The authors visited three classes once a week for 5 weeks; each session was approximately 30 minutes. They conducted pre- and post-evaluations to assess student interest in computer science and mathematics, and anxieties related to mathematics. One interesting result was that the students did not particularly favor the binary activity. This was the same response we received in our pilot, as we will discuss later.

Blum et al. [2] describe CS4HS (Computer Science for High Schools), an outreach program for high school teachers. The program was created to emphasize how computer science and computer science careers involve more than just programming. Workshop participants learned how to illustrate computer science concepts using CS Unplugged activities and other kinesthetic activities.

Closest to our work is that of Taub et al. [8], which focuses on analyzing the effect of CS Unplugged activities on participants' views of computer science. Their pilot program consisted of 13 seventh and eighth grade students. Eighteen CS Unplugged activities were presented. The evaluation consisted of a pre-survey focused on evaluating participants' views of computer science. (There was no mention of a post-survey.) Six students volunteered to participate in a structured post-interview. The interview involved viewing images and discussing their thoughts on what the images represented with respect to computer science. The authors conclude that CS Unplugged activities had a positive effect on students' views of computer science, but their effectiveness could be improved. One suggested improvement was to strengthen students' weak connection between the activities and computer science concepts being taught. The authors also note that the activities do not adequately represent the career opportunities in computer science.

3. PROGRAM DESIGN

Our outreach program consists of ten activity modules, nine of which are based on CS Unplugged activities [4]. The tenth is a computer architecture activity. We adapted each CS Unplugged activity to suit our class length, as well as the high school age range. Below is a short summary of each lesson prepared for this program.

Lesson 1: Binary Numbers. The goal of this module is to introduce students to the binary number system. We discuss reasons why it is necessary to use binary instead of decimal numbers to store information on a computer. We then introduce students to the idea of converting between different numbering systems and give them several examples, as well as a few conversions for them to solve on the board.

Lesson 2: The Anatomy of a Computer. The goal of this module is to demystify the innerworkings of a computer by teaching students about the constituent hardware components and their functions. With assistance from the instructors, the students disassemble and reassemble desktop computers in small teams. (This is the only lesson not taken from the CS Unplugged activities.)

Lesson 3: Information Theory. The goal of this module is to demonstrate the role of context in assessing the importance of elements in a collection. We open the activity by presenting a paragraph that demonstrates how the brain processes words while reading ("Aoccdrnig to rscheearch at an Eligngsh uinervtisy,..."). We discuss how our brain uses context clues to interpret information. We then presented activities that allowed the students to compare strategies for finding numbers in sorted and unsorted collections and explore the idea of finding data via binary search.

Lesson 4: Algorithms and Sorting. The goal of this module is to introduce students to algorithms and why algorithms are important to computer science. We explain the difference between two sorting algorithms (bubble sort and quicksort) using an activity involving sorting a group of students according to their height. We then discuss popular real-world applications of sorting, including iTunes track listings and Google's search results.

Lesson 5: Sorting Networks and Parallel Computing. The goal of this module is to introduce students to the idea of partitioning tasks to execute over multiple processing units. Using a job in a local grocery store, we draw a connection between having multiple employees at a job and multiple processors solving a computational problem. Next, we use the sorting network activity provided by CS Unplugged to demonstrate a parallelized sorting algorithm.

Lesson 6: Graphs and Minimum Spanning Trees. The goal of this module is to introduce students to the fundamentals of graph theory. We begin by drawing graphs describing airport routes and interstate maps between cities. We transition into the topic of minimum spanning trees and play a game using Prim's algorithm to find the cheapest way to tour Europe using a graph of major airports.

Lesson 7: Routing and Deadlock. The goal of this module is to introduce the concept of routing and the strengths and weaknesses of various network layouts. We discuss the need for redundancy, the possibility of deadlock when two cars come to a one-way bridge (with the CS Unplugged dining philosopher's activity), and the issue of privacy across networks (using a brief encryption / decryption exercise).

Lesson 8: Error Detection and Cryptography. The goal of this module is to introduce data corruption detection and encryption/decryption techniques. We use the CS Unplugged parity cards activity to explore how corrupted data could be detected and potentially fixed. We also share

data encryption techniques designed to safeguard information streams.

Lesson 9: Public Key Encryption. The goal of this module is to demonstrate public and private key encryption. To explain these concepts we use two activities. The first uses a locked box passed from a "host" student to a "destination" student through several "malicious" students. In the second activity, we teach students how to send encrypted messages and how to decrypt these messages.

Lesson 10: Programming Languages. The goal of this module is to introduce students to the types of instructions required to program a computer. The class is broken into two teams. Each team appoints a designated "robot". After creating a set of instructions, each group is tasked with navigating their blindfolded robot through a maze.

4. PILOT PROJECT

The pilot program spanned two semesters, targeting two sections of an introductory programming class at a local high school. Ten 40-minute activities were presented each semester.

4.1 Semester 1

In the first semester, our pilot followed a quasi-experimental, nonequivalent control group design. We were not able to control participant selection or account for baseline differences in student attitudes and content understanding. The experimental section included 14 students. A second section of 15 students served as the control group. Prior to the first activity, a pre-survey was administered to both sections. On the last day of the program, an identical post-survey was administered. To receive feedback on the instruction and activities, the schools' Instructional Coach[1] was asked to observe one of the class presentations.

Upon completion of the first semester, the pre- and post-surveys were evaluated. Our analysis, detailed in Section 5, indicated that the activities were not having the impact we were anticipating. Specifically, gains in interest in computer science and content understanding were insignificant. In response, we consulted the Instructional Coach and class instructor; the decision was made to add accountability to the program to insure the students were accountable for the lesson content.

4.2 Semester 2

In the second semester, the experimental group comprised the students that served as the control group in the first semester. For this reason, these students were not given another survey until the end of the semester. (Their post-survey from the first semester was considered as the pre-survey for the second semester.) These students were presented with the same activities as the first group. During each class, students were given a worksheet to complete containing 4 questions related to the lesson being taught. At the end of each lesson, the worksheets were collected, corrected, and returned to the students at the beginning of the following class. These worksheets had no impact on the students' grades. However, at the request of the instructor, two questions were provided each week to include in a weekly quiz. It was our hope that the addition of the worksheet and the quiz questions would add the layer of accountability needed for the program to be successful.

[1]An *Instructional Coach* works with teachers to provide regular feedback related to course standards, classroom management, standardized testing, and curriculum strategies.

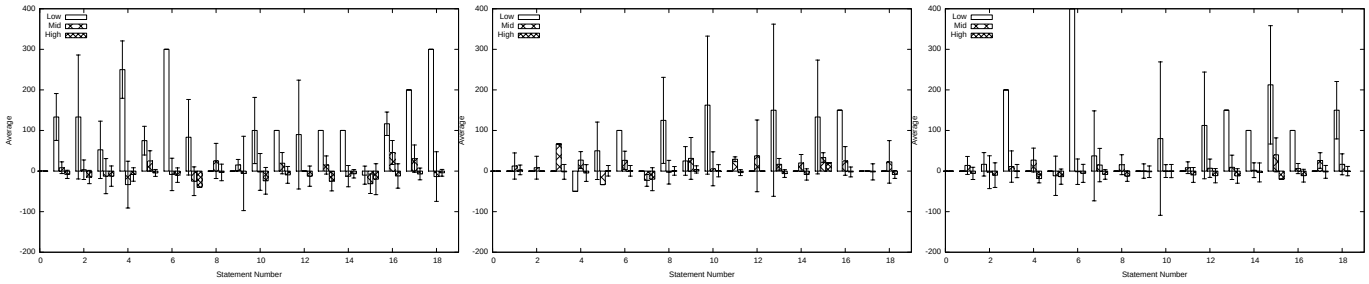
No.	Statement/Question
1	Computer science (CS) seems like it would be fun.
2	I might be interested in majoring in CS in college.
3	Working with computers is intimidating.
4	I think CS is useful in my daily life.
5	I can use a computer in math and science homework.
6	I am capable of doing well in a CS major in college.
7	CS is mostly about programming.
8	CS has a lot to do with math.
9	CS has a lot to do with problem solving.
10	CS would be useful to me even without a computer.
11	I think I understand how computers store information.
12	I would break something if I replaced a part in my computer.
13	I can guess a secret number between 1 and 100 in 10 guesses.
14	I think I understand how iTunes sorts songs.
15	I think I could explain what an "algorithm" is.
16	I think I basically understand how the Internet works.
17	I think I can find the tallest person among 16 in 3 minutes, with one comparison per minute.
18	In a list of 10 cities, I think it would be possible to reach the smallest city from every other city.
19	If you decided to pursue a CS degree in college, which topics or courses would you expect to study?
20	If you decided to pursue a CS degree in college, which high school classes do you think would be most important to prepare you?

Table 1: Student Survey

5. EVALUATION

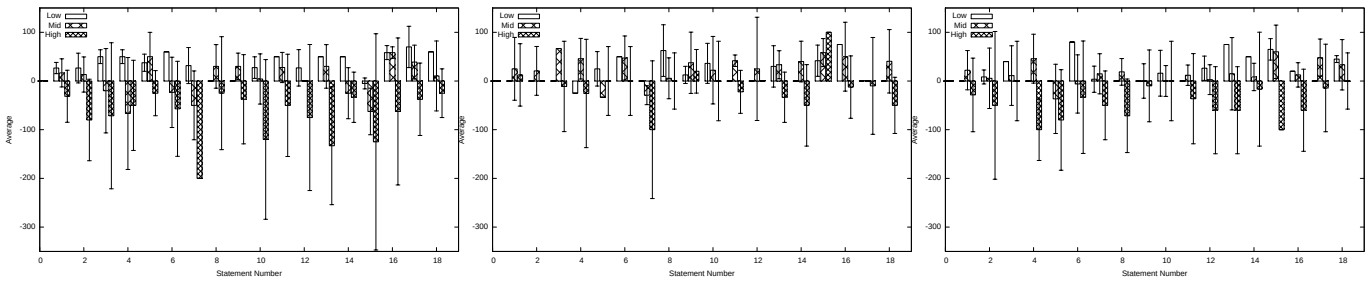
The pre-/post-survey consisted of 18 Likert-style attitudinal and content understanding statements, as well as 2 free response questions. Students were asked to rate their level of agreement with the statements, from *strongly disagree*, *disagree*, *moderately disagree*, *moderately agree*, *agree*, and *strongly agree*. Table 1 lists the survey statements and questions. Ratings of the statements were given the values from 1 to 6, for *strongly disagree* to *strongly agree*, respectively. When a statement had a negative connotation (e.g., statements 3, 7, 12), the values were inverted. Accordingly, an increase in the post-survey values indicates a desired result for all statements.

Rating Statements. A cursory review of each semester's survey results indicated little to no change in interest/confidence. To verify this, we performed a statistical analysis on the data from the first and second semester experimental groups. The analysis was to determine any significant change in the mean of the pre-/post- survey questions. First, for each question, a statistical F-test was performed to determine if the variance in the pre-/post- survey responses were equal. If the resulting p-value was greater than or equal to 0.05 (1 - confidence interval of 0.95), the variance was assumed to be equal. Second, for each question, a two-sample t-test was performed to determine if the pre-/post-surveys exhibited a significant change. Our original observations were verified in that only one question exhibited a significant change. Question 15 - "I think I could explain what an "algorithm" is." The first group had a pre-/post-survey mean of 2.73 and 4.72, respectively with a variance of 1.82 and 2.02, respectively. The F-test showed a p-value of 0.44, so a two-sample t-test with equal variance was performed. The p-value of the t-test was 0.01, indicating statistical significance. The second group showed similar results. The pre-/post- survey mean was 2.73 and 4.8, respectively



(a) Control (b) First Experimental (c) Second Experimental

Figure 1: Interest Gain



(a) Control (b) First Experimental (c) Second Experimental

Figure 2: Gap Closure

with a variance of 1.78 and 1.03, respectively. The F-test exhibited a p-value of 0.16, so again, a two sample t-test with equal variance was performed. The p-value of the t-test was 0.00005 indicating statistical significance. We hypothesize that this question showed significant change because the term "algorithm" was defined in an early module and revisited in many of the remaining modules. A worksheet with the complete set of data analysis can be found at: http://dsrg.cs.clemson.edu/PHSStatistics.xls.

Interest Gain and Gap Closure. We divided the student ratings for each Likert-style statement into three categories: *low* (1, 2), *mid* (3, 4), and *high* (5, 6) based on the pre-survey ratings. We wanted to study the effectiveness of our program on each group. For each group, we measured the percentage of interest/confidence gain using the following formula: [((post value - pre value) / pre value) * 100]. We also measured the interest/confidence gap closure in percent, measured using the following formula: [((post value - pre value) / (6 - pre value)) * 100]. A gap closure of -100 indicates a decrease of 1 point from a pre- to a post-survey value. Some categories have missing values, as no students rated the corresponding statements in these categories. Figures 1 and 2 summarize the average interest/confidence gain and average gap closure for each group, respectively. The x-axis represents the survey statement number; the y-axis indicates the average and standard deviation for the low, mid, and high groups. Notice that the averages for the *low* groups are consistently higher than the other groups. This indicates that we were more successful in increasing the level of interest/confidence among students in the low group than we were among students in other groups. We postulate that it is easier to raise the level of interest among students who are not interested in the subject than it is to raise the level of interest among students with high pre-existing interest. Notice from Figure 2 that the gap for the *high* category increased significantly for the second experimental group. This indicates that students gave lower ratings in the post-survey than they did in the pre-survey, thus showing less interest/confidence. However, most of these results remained in the high category, where students who scored 6 in the pre-survey scored 5 in the post-survey. To our disappointment, when compared with the control group, there was not a significant interest/confidence gain or gap closure among the two experimental groups, and in many cases, the results were negative, especially among the mid and high groups.

Free Response Question 1. The free response questions ask the students to identify the classes they expect to study in an undergraduate computer science program, and the high school classes they feel would prepare them for the major. We grouped the classes identified by the students (i.e., *"binning"*) and counted the frequency of each response in the pre-/post-surveys. For the first question we noticed the presence of four new topics (i.e., robotics, computer science, binary numbers, networking) in the post-survey for the first group. These were topics covered in our modules, but were not necessarily covered in their class. For the second group, we also noticed the introduction of computer science as a subject in the post-survey that occurred three times. Further, computer engineering decreased from four occurrences to two. The students seem to have learned that computer science and computer engineering are different disciplines. Finally, the data in both groups indicates the students seem to believe that computer programming is an important subject to study in computer science. The occurrence of this subject did not change for the first group, with four occurrences. However, the occurrence count went up with the second group from eight to nine. This was surprising, as we were teaching the students computer science concepts *without* the use of a computer, and we never emphasized the importance of computer programming in com-

puter science. We attribute this to the programming lessons they attended during the week. To our surprise, the control group seems to think that taking computer programming classes is not as important as they stated in their pre-survey. The occurrence count decreased from twelve to eight.

Free Response Question 2. As with the first question, we observed some interesting results concerning the second question ("... high school classes to prepare you for a computer science degree..."). We noticed that in the post-survey for the two experimental groups, the webpage design subject does not appear to be as important as students previously thought. For the first group, the occurrence count went down from one to zero, while it went down from three to one for the second group. Further, students seem to draw a stronger correlation between math and CS. The occurrence count went from one to three, and from one to four for the first and second groups, respectively. This differs from the control group results. Finally, both groups seem to think that taking computer classes or computer-related courses will help prepare them for a computer science major in college. The control group appears to draw a stronger correlation between CS and programming than the experimental groups, who instead seem to emphasize mathematics.

Discussion. We have shown that the CS Unplugged program we presented had, on average, no impact on student attitudes toward computer science or their perceived content understanding. We believe there are several causes for this result. First, based on our experience with presenting CS Unplugged activities to other groups, we believe high school students do not find kinesthetic activities of this kind as exciting as middle or elementary school students, such as the groups participating in [6–8]. Although we attempted to modify the activities to be more suitable for high school participants, we observed that these students seemed less motivated to participate than we had hoped. Second, the selected students were already in a computer science programming class. We felt they would be more interested in the activities. However, after reviewing the data and reflecting on our experiences with these students, we believe most of the students perceive themselves as "experienced programmers" and therefore may have been less interested in spending time learning these concepts. Third, students of this age group have already experienced the power and excitement of creating fun and rewarding programs, such as games. Accordingly, they may have viewed our activities as less rewarding and unnecessary. Last, most successful CS Unplugged-based programs were either summer workshops or after school programs. Participants in these programs tend to be self-selected, with a strong pre-existing interest.

Threats to Validity. It is worth emphasizing that a number of experimental factors were outside of our control. In future pilots, several validity threats must be addressed. This includes increasing the population size, adjusting for preexisting differences in student attitudes and content understanding, adjusting for instructor-induced impacts, understanding the impact of computing lessons external to our program, and addressing data collection failures (i.e., missing student response data).

6. CONCLUSION

Like most computer science educators, we are looking for new ways to excite the next generation of undergraduates about pursuing a degree in computer science. The CS Unplugged program appears to offer a promising approach. The basic tenets resonate with our own attitudes about computer science and teaching effectiveness, and the program is gaining adoption fast. Indeed, we believed that success was a foregone conclusion.

Unfortunately, we were wrong. Using a quasi-experimental, nonequivalent control group design, we evaluated the impact of a semester-long outreach program based on CS Unplugged, repeated for two consecutive semesters. The results indicate, with only one exception, that the program had no statistically significant impact on student attitudes toward computer science or perceived content understanding. Indeed, in some cases, it appears as though the program had an undesirable effect (although these results were not statistically significant).

The purpose of this experience report is not to attribute failure to the CS Unplugged program. We have presented an unbiased report of our approach to adapting the program to a high school context and identified possible explanations for the poor results. The intent is to engage the community in a discussion of how CS Unplugged should be used — *and how it shouldn't.*

Acknowledgments

This work is supported by the National Science Foundation through awards CNS-0745846, DUE-1022941, and DUE-0633506. Yvon Feaster is an NSF Graduate Research Fellow. The authors would like to thank the local high school administration and Mr. George Rosser, the class instructor.

7. REFERENCES

[1] ACM K-12 Task Force Curriculum Committee. A model curriculum for K-12 computer science: Final report of the ACM K-12 task force curriculum committee. csta.acm.org/Curriculum/sub/CurrFiles/K-12ModelCurr2ndEd.pdf, 2006.

[2] L. Blum and T. Cortina. CS4HS: an outreach program for high school CS teachers. In *Proceedings of the 38th ACM Technical Symposium on Computer Science Education*, pages 19–23, New York, NY, USA, March 2007. ACM.

[3] Bureau of Labor Statistics, U.S. Department of Labor. Occupational outlook handbook, 2010-11 edition, computer scientists. www.bls.gov/oco/ocos304.htm, 2010. (*date of last access*).

[4] Computer Science Unplugged. Computer science unplugged. csunplugged.org/, 2010. (*date of last access*).

[5] D. Heersink and B. Moskal. Measuring high school students' attitudes toward computing. In *Proceedings of the 41st ACM Technical Symposium on Computer Science Education*, pages 446–450, New York, NY, USA, March 2010. ACM.

[6] L. Lambert and H. Guiffre. Computer science outreach in an elementary school. *Journal of Computing Sciences in colleges*, 24(3):118–124, 2009.

[7] C. Mano et al. Effective in-class activities for middle school outreach programs. In *Proceedings of the 40th ASEE/IEEE Frontiers in Education Conference*, pages F2E1–F2E6, Washington DC, USA, October 2010. IEEE.

[8] R. Taub et al. The effect of CS Unplugged on middle-school students' view of CS. In *Proceedings of the 14th ACM Annual Conference on Innovation and Technology in Computer Science Education*, pages 99–103, New York, NY, USA, July 2009. ACM.

[9] J. Vesgo. Enrollments and degree production in US CS departments drops further in 2006–2007. *Computing Research News*, 20(2):4, 2008.

[10] S. Zweben. Computing degree and enrollment trends from the 2008-2009 CRA Taulbee survey. www.cra.org/govaffairs/blog/wp-content/uploads/2010/03/CRATaulbee-2010-ComputingDegreeandEnrollmentTrends.pdf, 2010.