A New CS Curriculum: Bridging the Gap Between In-Class Assignments and Real World Problems

A Technical Report submitted to the Department of Computer Science

Presented to the Faculty of the School of Engineering and Applied Science
University of Virginia • Charlottesville, Virginia

In Partial Fulfillment of the Requirements for the Degree
Bachelor of Science, School of Engineering

Ethan Choo
Spring, 2021

Technical Project Team Members
Anita Ho
Edward Cheng
David Wang

On my honor as a University Student, I have neither given nor received unauthorized aid on this assignment as defined by the Honor Guidelines for Thesis-Related Assignments

Signature _____ _____ Date 4/26/2021__
Ethan Choo

Approved _____ Date _____
Seongkook Heo, Department of Computer Science

Approved _____ Date _____
Haifeng Xu, Department of Computer Science

A New CS Curriculum: Bridging the Gap Between In-Class Assignments and Real World Problems

Edward Cheng
University of Virginia
ec8bhb@virginia.edu

Ethan Choo
University of Virginia
ec4kj@virginia.edu

Anita Ho
University of Virginia
ahh2re@virginia.edu

Yuyang Wang
University of Virginia
yw2aj@virginia.edu

ABSTRACT

This project focuses on developing a course that covers technical and behavioral interview questions that are often asked of computer science students when hiring for internships or full time jobs. Oftentimes, students have difficulty using what is taught and done in class to develop appropriate responses to interview questions. Currently, there is not much research on this topic as most research focuses on just the interview aspect, and so there is little actual research on how CS courses prepare students for interviews. Thus, this project develops ways for students to be better prepared for computer science interviews. Alternatively this project shows how CS courses are not able to prepare students adequately for interviews. Research is done by looking at existing interview questions from companies hiring CS students and by looking at the assignments and lectures that the CS courses provide. Additionally, hiring managers were interviewed to identify what things are looked for in interview responses.

After testing the curriculum on seven undergraduate fourth year UVA students, their technical interview correctness and time to code significantly increased. In addition, they were able to articulate their contributions in previous software development projects to the interviewer and display qualities that are looked for by hiring managers in the behavioral interview. The implications of our research show that current CS courses are lacking since they don't bridge the gap between in-class material and real world applications.

INTRODUCTION

Students often ask others what questions they were asked in their recent interviews, or what questions do certain companies ask. These questions reveal a fundamental misunderstanding of where questions come from and a lack of knowledge of the tech interview process. At a large majority of large tech companies, there are no question lists for what interviewers should ask, but instead, each interviewer picks their own questions. Since it'"s somewhat of a "free for all" as far as what questions are asked, there's nothing that makes a question a "recent Google interview question" other than the fact that some interviewer who happens to work at Google just so happened to ask that question recently. The questions asked this year at Google for all intents and purposes do not really differ from those asked three years ago. In fact, the questions asked at Google generally don't differ from those asked at similar companies (Amazon, Facebook, Netflix). For the most part there are some broad differences across companies. Some companies focus on algorithms, while others focus more on knowledge-based questions. But within a given category of question, there actually is kind of little that makes it "belong" to one company instead of another. A Google algorithm question is essentially the same as a Facebook algorithm question at a high overarching level. [2].

Another common question is "How will you be evaluated?" Interviewers assess your performance relative to other candidates' performances on the same question by the same interviewer. Your interviewer develops a feel for your performance by comparing your performance relative to other people's performances. It's not about the candidates he/she has interviewed that week. It's about all the candidates that he/she has ever asked this question to. While most people complain when they get a hard question, for this reason, it's not a bad thing [2]. These gaps in knowledge illustrate very important concepts to understand while interviewing with tech companies, whether it be for an internship or full time position. Many curriculum classes do not explicitly cover these subjects, putting the students and candidates at a disadvantage. Having a class dedicated to tech interviews can largely benefit the majority of computer science majors, as most of them will be seeking employment at large tech companies. Thus by preparing them for what to expect, and giving them the proper practice to demonstrate their knowledge and present the best side of them to interviewers, students will more reliably be able to land jobs in computer science.

In this project a four-week curriculum designed to prepare students for technical and behavioral interviews by bridging the gap between the current computer science coursework and real world application. The goal for students for the technical portion of this course is to recognize patterns in interview questions, use algorithms to solve coding problems, and be able to explain their code and understand its time and space complexity. The goal for the behavioral portion is to be able to describe their contributions, challenges, and failures on previous projects, recognize what qualities interviewers are looking

for, and to prepare responses to typical behavioral interview questions.

BACKGROUND

At the University of Virginia, current students and recent graduates in the computer science department found that many of the technical interview questions had concepts that were taught in primarily CS 4102 Algorithms and CS 2150 Program and Data Representation, while course material in CS 3240 Advanced Software Development Methods gave interviewees content to talk about during the behavioral portion of the interview. In Figure 1, a distribution of topics that Microsoft asks during interviews is illustrated. The figure makes it apparent that in terms of actual concepts asked during interviews, students at the University of Virginia should not have any trouble passing interviews at top companies such as Microsoft, Amazon, Facebook, and Google. Thus the problem comes from either how the concepts are being taught or how students are preparing for interviews.

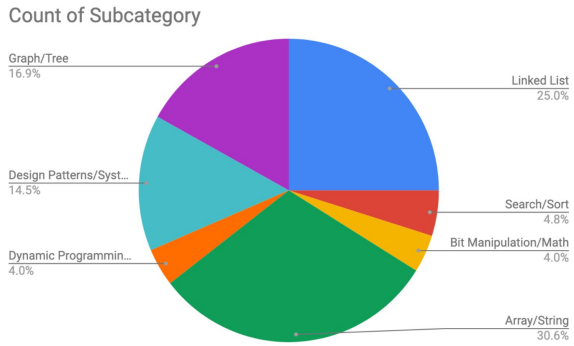

Figure 1: The figure above shows a distribution of categories of percent frequency asked during an interview.

Additionally, students and graduates said that they often studied using outside material and would study at the last minute right before interviews in order to be fully refreshed. Thus, this paper and associated project looks at specifically how information from UVA courses could be combined and taught to students at the University of Virginia in order to help students pass interviews more consistently and with less last minute preparation.

RELATED WORK

As the tech industry has grown, so has the number of job opportunities and the number of people studying computer science. This has resulted in a huge space for interview prep as schools traditionally do not have any classes explicitly for interview prep. Some of the largest resources interviewees use to prepare are Leetcode and Cracking the Coding Interview. These two have become the primary recommendations for people wondering what to use to study. Leetcode is a free website with sample interview coding questions, an editor to compile code, and a discussion board [3]. It also has other features locked behind a premium membership such as full code solutions and explanations. Cracking the Coding Interview, on the other hand is a book that walks you through the interview process with information about how interviews are conducted, steps to prepare, and sample questions and answers [2]. There are also other resources such as Youtube and other websites, either free or paid, such as, Hackerank, CareerCup, etc. Additionally, UVA offers mock interviews through the Career Center and other programs, however, these programs are not specialized to the CS curriculum and thus will still result in students being underprepared when attempting to complete technical interviews and coding problems during interviews for CS jobs. These resources often only offer very limited support, and many students may need additional attention and help to enable them to succeed. This is where this class comes in.

SYSTEM DESIGN

The course is designed to introduce students to common interview questions and help students develop strategies to solve interview questions. The material in this course will work to integrate concepts from both CS 3240 Advanced Software Development Methods and CS 4102 Algorithms in order to prepare students for a career in the software development field as well as illustrate how material learned from CS 4102 Algorithms can be applied to more practical, hands-on classes such as CS 3240 Advanced Software Development Methods. Although this course focuses on preparing students for a career in software development, strategies and general concepts can be applied to other jobs in the CS field.

The goal for students who complete this course successfully will be able to recognize patterns in interview questions, use strategies to solve the problem, and be able to explain their code and understand the time complexity. To this we will be teaching CS 6969 as a lecture based style class with weekly meetings on Tuesday and Thursday 2:00-3:15 pm. Although students will not be graded based on participation, it is highly encouraged to interact and ask questions to further the understanding of the material. Students are also encouraged to speak with the instructors about any concerns about classroom dynamics and the course in general. The students progress and mastery will be done through exams, which will be given through mock-interviews with the students through the TAs. The exams will include both behavioral and technical portions.

For the first week in the first class of the week, there will be assigned readings in the book. We will cover

strategies to approach array problems, and go over examples where the specific strategies fit well. The second class of the week will mostly be going over thought processes to approaching the problem through examples. Classes will encourage random student participation and volunteers for ideas on how to solve the problem. Take-home assignments will involve a list of problems and students must not only solve it, but identify their approach towards and what strategies worked well and what didn't. An example problem we would go over is as follows:

> There is a server consisting of a very large matrix of 0s and 1s. Many clients want to know how many 1s are within a rectangular area given the corner points. You may do anything to prepare your server before the client queries begin. Once the client queries begin, your server is expected to be able to output the results fairly quickly O(1).

For the second week in the first class of the week, there will be assigned readings in the book. We will cover strategies to approach string problems, and go over examples where the specific strategies fit well. The second class of the week will mostly be going over thought processes to approaching the problem through examples. Classes will include random student participation and volunteers for ideas on how to solve the problem.

Take home assignments will involve a list of problems and students must not only solve it, but identify their approach towards it and what strategies worked well and what didn't. Homeworks will include a list of questions related to the current topic, but also review questions from previous topics. An example problem we would go over is as follows:

> There is a newly invented magical function called "isSubstring(String a, String b)" that returns a boolean True of False if String a is a substring of String b. This magical function has a time complexity of O(1). Using this function, write another function called isRotation(String a, String b), that returns True if String a is a rotation of String b.

For the third week in the first class of the week, we will have pre-assigned readings in the book. We will cover strategies to approach mathematics-computer science problems, and go over examples where the specific strategies fit well. The second class of the week will mostly be going over thought processes to approaching the problem through examples. Classes will include random student participation and volunteers for ideas on how to solve the problem. Take home assignments will involve a list of problems and students must not only solve it, but identify their approach towards it and what strategies worked well and what didn't. Homeworks will include a list of questions related to the current topic, but also review questions from previous topics. An example problem we would go over is as follows:

> There is a 3 dimensional room where all the walls are mirrors. Given the angle (rho and theta) a beam of light enters the room from a specific location, what is the first corner of the room that the light perfectly touches.

For the fourth week in the first class of the week, we will have pre-assigned readings in the book. We will first explain how behavioural interviews are "graded". An important concept here is to explain that passing an interview is reliant on how the interviewee is perceived by the interviewer. Thus emphasis will be placed on strategies that will help students clearly communicate ideas such that the interviewers/graders can understand. Interview strategies such as the STAR method will be taught. The second class of the week will mostly be going over thought processes to approaching the problem through examples. Classes will include random student participation and volunteers for ideas on how to solve the problem.

Take-home assignments for this week will involve a few behavioral problems that students must write appropriate responses to. In addition to this, students will have to identify what the thought process was when approaching each of the behavioral problems. Additionally, for this week students will think of a project that they have worked on in the past that could be used for the STAR method and could be talked about for a sample behavioral problem. An example problem we would go over would be, "Tell me about a challenging interaction with a teammate?" Since behavioral questions are less cut and dry as the technical questions, we would look for answers with components of a good answer, such as:

> Situation: During my CS 3240 Advanced Software Development Methods project, I was working with three other members. Two of the members were very interactive during meetings, communicated well, and got their work done on time, however, the last member did not speak much and did not regularly complete the work we assigned them.
> Task: The task was to be able to complete the project. With the current situation, it would be difficult to complete the project if one of the members was not doing their share of the work.
> Action: In order to understand why this group member was not contributing as much, I reached out to them and asked if they had any personal issues going on at the time. This resulted in them telling me that they did not feel confident with

their coding abilities and feared they would mess up the project.

Result:After explaining the basics of how they could go about contributing to the project without worrying about breaking any major components, they were more comfortable taking on more tasks.

PROCEDURE

The deliverable of this project is a four-week curriculum designed to prepare students for technical and behavioral interviews by bridging the gap between the current computer science coursework and real world application. The goal for students for the technical portion of this course is to recognize patterns in interview questions, use strategies and algorithms to solve the problem, and be able to explain their code and understand the time and space complexity. The goal for the behavioral portion is to be able to describe their contributions, challenges, and failures on previously done projects, recognize what qualities interviews are looking for, and to prepare responses to typical behavioral interview questions.

The course is designed to be administered in a way similar to current UVA courses, either twice a week for 75 minutes or three times a week for 50 minutes. There are detailed lecture topics for the professor or lecturer to discuss each class as well as homework assignments and a course grading rubric. There is also a before and after behavioral and technical interview component and a post-course survey to gather information on how much the students benefited from the course.

RESULTS

The four-week curriculum was tested by seven fourth year undergraduate computer science majors attending the University of Virginia. These students were actively searching for internships and full-time positions in the full-stack of computer science. To evaluate the benefit of the curriculum, a practice technical and behavioral interview was administered before and after taking the course. Additionally, a survey was given to each participant to judge their opinion on how much the curriculum helped them and whether it was worth their time.

Two practice behavioral and technical interviews were created to test their improvement over the course. The behavioral interview grading rubric was based on the following qualities displayed in their previous projects: challenges, failures, enjoyment, leadership, conflicts, and what they would have done differently. The results from behavioral interviews show significant improvement across all six of these areas for the participants in the study. According to the survey given afterwards, knowing what the interviewer is looking for during the behavioral interview was what they benefited from the most from the curriculum. Additional things that were found to be helpful to the applicants for the behavioral interview were organizing their projects and previous work, preparing and practicing projects to talk about, and receiving feedback on their answers. Although the qualities in the rubric were the most common qualities interviewers are looking for, not every interviewer is looking for these exact same qualities during a behavioral interview. This is one shortcoming of the curriculum developed.

Each of the two technical interviews were created by taking a variation of an easy, medium, and hard question from previously done interviews from Facebook, Google, and Amazon and questions found on Leetcode. The technical interviews were given so that half of the participants took the first interview before the curriculum and the other half took the other interview. After the curriculum, the participants took the respective remaining interview to make up for any differences in difficulty between the questions in the two interviews. Factors that were considered in the interview were correctness, spatial complexity, time complexity, and time spent coding. The time spent coding decreased by 36 percent after the curriculum. The spatial and time complexity of their algorithms did not improve in over 90 percent of the problems. However, correctness of dynamic programming and tree/graph questions increased significantly after the curriculum from 22 percent to 68 percent. The overall correctness of medium questions improved from 29 percent to 86 percent. The overall correctness of easy and hard questions increased by 14 percent and 29 percent, respectively. Lastly, after the curriculum, during the interview, participants were more likely to identify and discuss the type of problem as well as the name of the algorithm, if any. According to the survey, most of the participants found dynamic programming practice to be the most useful and commented that they did not understand it as well when they first learned it in their Algorithms class. In addition, they found teaching by example through practice problems to be the most effective learning method.

The survey found CS 4102 Algorithms to be the most useful class in technical interview preparation. Participants recall learning about the topics in their coursework. However, in many cases, they were unable to apply the information to the technical question because they couldn’t remember the exact steps to the algorithm. The class that students found the most useful for behavioral interviews was CS 3240 Advanced Software Development as they have live experience working on a full-stack development project that they were able to discuss during the interview.

CONCLUSIONS

The four-week curriculum created in this project focuses on developing a course that covers technical and behavioral interview questions that are often asked of

computer science students when hiring for internships or full time jobs. Oftentimes during interviews, students have difficulty applying what is taught in class to develop appropriate responses. There is currently little research on the gap between the computer science curriculum and software development interview. Thus, this project develops ways for students to be better prepared for computer science interviews. Alternatively, this project shows how CS courses are not able to prepare students adequately for interviews. A four-week curriculum was created by comparing existing interview questions to assignments and lectures from current computer science courses. The main teaching focus was to create a set of questions that encompassed every topic in technical interviews and teach through example and practice. Additionally, hiring managers were interviewed to identify what things are looked for in interview responses. The main qualities from behavioral interviews were teamwork, challenges and failures, enjoyment, conflict resolvement, and growth mindset. The results from a trial curriculum given to seven fourth-year undergraduate computer science students showed significant improvement in both the technical and behavioral interviews. Additionally, the participants recall learning about the topics in their computer science coursework but originally found it difficult to apply it to technical interview questions. For the more difficult questions on the technical interview, participants did not recall the algorithm used to solve the problem but remember the topic being discussed. In regards to the behavioral interview, participants were unsure what interviewers were looking for and found their computer science coursework not beneficial to the behavioral part. These results show that current CS courses are lacking since they don't bridge the gap between in-class material and real world applications.

FUTURE WORK

We only had time to design four weeks worth of material, but if there was more time, we would have designed a full curriculum that covered other popular topics such as trees, graphs, searches, sorts, linked lists, dynamic programming, and system design. These topics can all be taught in a similar manner as the three technical subjects, arrays, strings, and math, we have already designed. We would also look to design a final to synthesize all the topics together. It is easy to look at a problem and solve it given a strategy, but once the students have learned many strategies to cover a wide variety of problems, it is difficult to choose which strategy is best for each scenario. We would have also liked to have more time to test our teaching methods on more people to optimize the class and how we help and support students in their interview preparation.

This interview preparation course design can also be expanded to other majors with similar interview processes such as systems engineering and many other engineering majors. Systems engineers often go into consulting where the interviews have very similar components. There are behavioral questions and cases, which would be equivalent to the computer science technical questions.

REFERENCES

[1] "How To Pass the Microsoft Interview: A Practical Guide." Byte by Byte, 15 Jan. 2021, www.byte-by-byte.com/microsoft-interview/.

[2] McDowell, Gayle Laakmann, 1982-. Cracking The Coding Interview : 150 Programming Questions and Solutions. Palo Alto, CA :CareerCup, LLC, 2011

[3] "The World's Leading Online Programming Learning Platform." LeetCode, leetcode.com/.