Preliminary Analysis of the 2014 Snapshot of SUNY CS Gatekeeper Courses

Sen Zhang,
Department of Mathematics, Computer Science and Statistics, SUNY Oneonta,
Ph: 6074363382
Zhangs@Oneonta.edu

Cynthia Marcello
Science, Technology, and Math Division,
SUNY Sullivan,
Ph: 845-434-5750 ext. 4322,
cmarcello@sunysullivan.edu

Arthur Hoskey
Computer Systems,
School of Business,
Farmingdale State College,
Ph: 631-420-2435,
hoskeya@farmingdale.edu

Howard Reed
Business and Information Technology Department
SUNY Delhi
Ph: 607-746-4109
reedhe@delhi.edu

James Antonakos
Computer Science Department
Broome Community College
Ph: 607-778-5122
antonakosjl@sunybroome.edu

ABSTRACT

Until K-12 fully implements computer science (CS) education nationwide, Higher Education Institutions (HEIs) must continue to assume an active, effective, and remedial role in bridging incoming freshmen to computer science. This paper discusses a fundamental and focused study that was conducted on the gatekeeper course offerings in CS programs/departments and computing-relevant fields in fall 2014 semester at SUNY system. In particular, we analyzed a snapshot of the SUNY CS gatekeeper courses. We investigated the spectrum of gatekeeper courses, recognized the diverse needs of each, classified them into categories, and studied various aspects of the courses including their design goals, the topics of the courses, and characteristics of the courses. SUNY practices find matches from nationwide HEIs for all categories. We conclude the paper by sharing observations, thoughts and discussions on the current situations from a holistic or systemness perspective.

INTRODUCTION

CS education at the higher education institution (HEI) level is already challenging on its own. CS curricula show great diversity, varying from campus to campus (ACM/IEEE-CS 2013). Compared with Math, Physics, and many other fields, computer science (CS) is a still young, vibrant, fast-evolving and fast-expanding field. For this study, we use CS to refer to a broad range of computer-related fields including existing and emerging fields such as computer science, information technology, management information technology, computer systems, software engineering, etc.

To further compound the issue, a CS component is still missing from the K-12 curriculum in NY and many other states (Wilson et al. 2010, Crawford 2013). Outgoing HS graduates with weak exposure to CS become incoming college students and continue to be a challenge for HEIs in terms of CS curriculum.

As a result, CS gatekeeper courses at HEIs need to be designed with proper and sensitive consideration of the above two factors. Gatekeeper courses here refer to college-level courses directly interfacing with High School curriculum, typically taken as the first course by college students as a general education or the first degree course. A

good start is half of the success. We believe gatekeeper courses, despite their introduction and remedial nature, are crucial because they will affect all the higher-level courses on the curriculum hierarchy and all the students entering into the program. This is more important for CS as opposed to other fields where students generally receive adequate exposure in their K-12 studies.

To help better understand the CS gatekeeper courses offered at SUNY, the authors of the paper, under the support of a SUNY IITG grant, conducted a project to explore a semi-standardized concept based Introduction to Computer Science course. As part of the work of the project, we surveyed the spectrum of the gatekeeper courses within SUNY and we attempted to categorize them. For each category, we also found matches from computing programs at the national level.

METHODOLOGY

We visited all SUNY school websites, scrutinized the publically available course offering information, decided on one gatekeeper course for every school in the system (assuming they had one) and compiled them in an Excel spreadsheet. We focused on the gatekeeper courses, typically at the 100-level or the corresponding numberings. The information collected include the school name, degree level of the program the course is a part of, course title, course description, course type, language used, and course number (Arthur et al.,2014). We always looked for a Computer Science program first. Specifically, if a Computer Science program exists for a SUNY campus then we took the gatekeeper course from that program. Otherwise, we used another Computer relevant program, for example Computer Information Systems (CIT) or Computer Systems etc. It is worth noticing that it is not atypical for a SUNY campus to have multiple programs related to CS. For example, Computational Science and Computer Science co-exist in SUNY Brockport. Some of the courses are part of multiple programs, given the interdisciplinary nature of the program. Some of the decisions required more investigation than we initially expected. For example, a first CS course is listed in the program curriculum requirements but that course has another CS course (not listed in the requirements) as a prerequisite. In that case we chose the prerequisite as the gatekeeper course. We did this because it seemed like the prerequisite course would have to be taken by most students. We claim this is a preliminary analysis of a curriculum snapshot, because we are aware the following: 1. The CS curricula keeps evolving. 2. We only select part of the curriculum. 3. Due to space limit, we have to omit many other discussion.

RESULTS

There are 64 campuses throughout the SUNY system. 57 of the SUNY schools offer some sort of gatekeeper course. The focus of these gatekeeper courses varies across the SUNY system. For the purposes of this study, the gatekeeper course that was required for the major or minor was chosen to represent the program. In a handful of cases there was more than one gatekeeper-like course that students needed to take so one was chosen. The majority of the gatekeeper courses (54%) focus on teaching students programming skills. A computer science overview is only taught in 21% of the courses. The other course offerings focus on information technology (IT), applications, web programming, and computer graphics. As evidenced by this data, there is a lack of standardization for

gatekeeper courses throughout the SUNY system. In addition, there is variety between the programming courses themselves since they are taught in different programming languages. As a result of these discrepancies, the gatekeeper course credits can be hard for students to transfer between schools. Furthermore, even if the student is able to transfer a course they may have difficulties when they move on to higher-level computing courses. The breadth and depth of the subject matter and the programming language that is taught may differ even though the course was transferable. Furthermore credit hours vary from program to program, ranging from 1 to 4.

Based on our analysis of the available course catalogs and course syllabi of the surveyed SUNY CS gatekeeper courses, we found that SUNY gatekeeper courses have been thoughtfully designed by the discipline faculty with one or more of the following attributes taken into consideration: informational, objective, engaging, fundamental, inspirational, recruiting, retaining, broadening, servicing, palatable, objective, remedial, bridging, and challenging.

CLASSIFICATION

One way to categorize the CS Gatekeeper courses would be either CS 1 or CS 0, under the common assumption that CS 1 is the first CS core course and CS 0 is a remedial course aiming to make up the missing step before the CS 1 course. However, no national standard has been setup for either of them regarding the content of the courses. For example, there is no agreement that CS 1 is Programming 1. As for CS 0, the adoption of a CS0 course by HEIs on a nationwide scale remains to be unclear. On campuses where various CS0 courses are offered, there is still no consensus on curriculum standards in terms of student learning outcomes and topics. CS0 has been a recurring theme for years at CS education conferences (Cook 1997, Hickey 2004, Gray et al. 2012). To further compound the issue, a CS 0.5 course using a media computation approach has also been reported (Sloan 2008) to engage students with diverse backgrounds. Also considering that most SUNY campuses do not explicitly adopt CS 0 or CS 1 to name their courses and many courses at this level are designed for general education;, we instead use our own categorization method to highlight the big picture of the diversity of the gatekeeper courses and gain a clear aggregated view of the surveyed gatekeeper courses, as shown in Figure 1.

We categorized each course into one of the following four categories: Programming, CS Overview, Applications (Microsoft Office Suite as a typical one), and others (combining IT, Web, Computer Graphics. etc.). We then counted up the number of courses for each of these categories as well as a percentage makeup. We also count the number of courses that are part of the different degree types. For each type, we pick a representative curriculum from SUNY campus, and briefly present its academic assumption on the student side, advantages and disadvantages, and its position in the CS program curriculum at that campus.

1. Programming. Expecting a coding-readiness of the ideal CS students. It can be further divided into two categories.
    1.1 Coding intensive using C++ or Java. Purely CS core with an expectation to excite students and truly learn the language. Assuming strong CS, math, logical and critical thinking skills from students. A representative is BCS 120 Foundations of Computer Programming I (in C++) offered by Farmingdale State College.

1.2 Coding moderate using easier or more engaging languages than C++ or Java, like Python, Javascript, etc. Use the language as a vehicle to make students feel programming is interesting. Usually for recruiting or combined purposes. A representative curriculum is CS 110 Programming Concepts and Applications (in Python) offered by Binghamton University.

2. Overview. Informational and concept-based, typically providing a fundamental foundation. A representative curriculum is CISS 100 Introduction to Computing and Information Sciences Hudson Valley Community College.
3. Application (Microsoft Office Suite, as a typical example). A typical service course, semi-general education and sometimes part of an Information Technology related major or minor. A representative is CIS 130 Productivity Computing offered by North Country Community College.
4. Others. Do not belong to any of the above. Typically, Web Development, Computer and Society, or Digital Literacy etc. will fall in this category. These courses generally cater to the learning needs of general education. A representative SUNY curriculum is CSC121 Introduction to Computing and the Web offered by State University College at Plattsburgh.

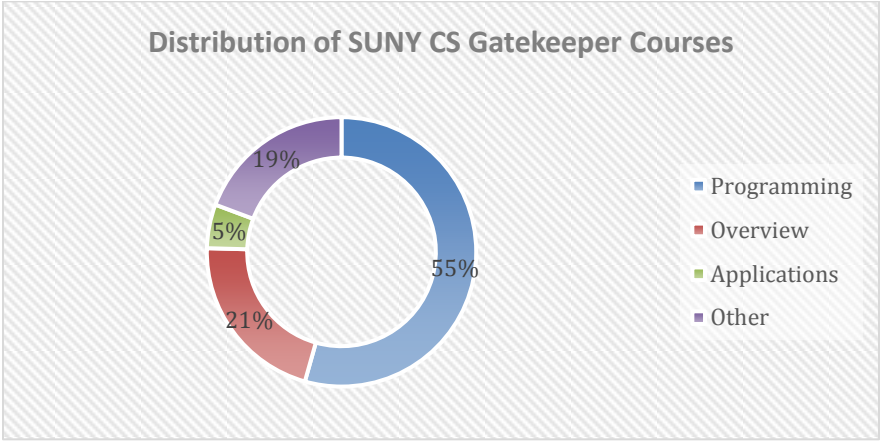

Figure 1: Categories and Distribution of CS Gatekeeper Courses

We also extended our analysis to outside SUNY and for each type we were able to find some representative curricula (Table 1).

Table 1: Representative Curriclua at Nationwide Scope

	Category	Institution Name	Course Name
1.1	Intensive Programming	Harvard	CS50 Introduction to Computer Science (Harvard)
1.2	Moderate Programming	MIT	6.189 A Gentle Introduction to Programming Using Python (MIT)
		Berkeley	CS0 : The Beauty, Joy and Awe of Computing (Clancy)
2	Overview	Stanford	101 Computer Science (Standford)
3	Application, Office Suite	Shepherd University	CIS 102 Microcomputer Apps (Shepherd)
4	Others	University of North Carolina at Charlotte	eScience (NCC)

DISCUSSION and OBSERVATION

Our initial studies and analysis reveal and confirm that inside SUNY the CS gatekeeper courses vary widely from campus to campus. But as a whole, the overall CS gatekeeper course offerings are consistent with them at HEIs nationwide. This shows that the SUNY system as a whole is consist with the overall state of CS education nationwide in terms of the diversity of the gatekeeper courses.

Individually, every approach is valid and contains an element of truth. However, there is no single recipe working for all campuses. Depending on the typical clientele a program admits, the gateway courses might fall into a variety of categories. For example, Harvard has their students start with C++ or Java. This type of intensive programming course, without an introductory course, may discourage less prepared students, those typically found in community colleges and 4-year liberal arts, from pursuing further courses or majoring in the field. These courses could be totally inaccessible to typical community college students, who tend to have a lack of previous experience and preparation. Sometimes, a program may offer different types of gatekeeper courses to serve the learning needs of different types of students (self-motivated, at-risk, students with CS AP, students going to Ivy leagues etc.).

Collectively, though, a clear disadvantage of a wide spectrum of gatekeeper courses is that they have created issues with understanding the fields for first time learners and make seamless transfer (both SUNY and nationwide initiatives) harder than other fields where gatekeeper courses are relatively standardized across the board.

Based on the above analysis and acknowledging that K-12 is still missing a CS curriculum and will continue to miss it in the near future, we believe that for most typical liberal arts colleges the gatekeeper courses need to be fundamental, informative and remedial and to be programming language independent. For those universities like SUNY Stony Brook or Binghamton, who typically get well-prepared students, a course of the above feature will still be valid, because not all of their incoming students will have CS AP. Options do exist for this type of remedial courses to be challenged or waived based on proper CS AP scores of students.

The SUNY system, the largest HEI in the nation, has a unique advantage to explore a semi-standardized gatekeeper CS course with the possibility to be a more transferable model for this type of courses than before across SUNY and HEIs in general. Due to page limit, we refer interested readers to the SUNY IITG project website (Zhang, 2014) to obtain the latest state of the progress.

ACKNOWLEDGEMENT

This work is partially supported by SUNY IITG 2014. The authors also want to acknowledge the SUNY system, and the SUNY campuses and many colleagues of the authors for their support.

REFERENCES

ACM/IEEE-CS Joint Task Force on Computing Curricula. (2013). Computer Science Curricula 2013. ACM Press and IEEE Computer Society Press.

Clancy, M., Garcia, D., & Harvey. B. (n.d.). The Beauty, Joy and Awe ofComputing. Retrieved from http://www.eecs.berkeley.edu/news/cso.pdf.

Cook, C. R. (1997). CS0: computer science orientation course. ACM SIGCSE Bulletin, 29(1), 87-91.

Crawford, C. (2013, November). Computer Science Principles and AP Computer Science A To Count For Mathematics Requirement In Alabama. Retrieved from http://cs10kcommunity.org/blog/csp-as-math-alabama.

Gray, J., Abelson, H., Wolber, D., & Friend, M. (2012, March). Teaching CS principles with app inventor. In Proceedings of the 50th Annual Southeast Regional Conference (pp. 405-406). ACM.

Harvard (n.d). CS50. https://cs50.harvard.edu/

Hickey, T. J. (2004, March). Scheme-based web programming as a basis for a CS0 curriculum. In ACM SIGCSE Bulletin, 36(1), 353-357.

Hoskey, A., Zhang, S., Marcello, C., Reed, H. & Antonaks, J. http://employees.oneonta.edu/zhangs/iitg2014/csgatekeeper_survey.html

Looby, G. James (n.d.). Retrieved from www.ciss100.com

Maloney, J., Burd, L., Kafai, Y., Rusk, N., Silverman, B., and Resnick, M. (2004). Scratch: A sneak preview. Second International Conference on Creating, Connecting, and Collaborating through Computing, 104-109. Kyoto, Japan.

McGettrick, A., Roberts, E., Garcia, D. D., and Stevenson, C. (2008). Rediscovering the passion, beauty, joy and awe: making computing fun again. In Proceedings of the 39th SIGCSE Technical Symposium on Computer Science Education (Portland, OR, USA, March 12 - 15, 2008). SIGCSE '08. ACM, New York, NY, 217-218.

MIT (n.d). CS6.189. Retrieved from http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-189-a-gentle-introduction-to-programming-using-python-january-iap-2011/

Shiva Azadegan, Josh Dehlinger, and Siddharth Kaza. (2014). Revitalizing the computer science undergraduate curriculum inside and outside of the classroom using mobile computing platforms (abstract only). In Proceedings of the 45th ACM technical symposium on Computer science education (SIGCSE '14). ACM, New York, NY, USA, 713-713.

Stanford (n.d.). CS101, Retrived from https://www.coursera.org/course/cs101

Shepherd University. CIS102. http://catalog.shepherd.edu/preview_program.php?catoid=5&poid=218&returnto=195

Sloan, R. H., & Troy, P. (2008, March). CS 0.5: a better approach to introductory computer science for majors. In ACM SIGCSE Bulletin, 40 (1), 271-275.

NCC. University of North Carolina at Charlotte. eScience. http://cciweb.uncc.edu/ ~mirsad/ITIS%201350/ITIS%201350.htmv

Wilson, A., Sudol, L., Stephenson, C., and Stehlik, M. (2010). Running on Empty: The Failure to Teach K-12 Computer Science in the Digital Age. ACM and CSTA. Retrieved from http://www.acm.org/runningonempty/roemap.html.

Wilson, B. C. (2002). A study of factors promoting success in computer science including gender differences. Computer Science Education, 12(1-2), 141-164.

Zhang, S., Hoskey, A., Marcello, C., Reed H., & Antonaks, J. SUNY IITG Semi-Standardarizing CS project website http://employees.oneonta.edu/zhangs/iitg2014/