Data Exploration and Visualization

Introduction

The need for this exploratory analysis is anything but nominal. There has never been a better time for students to pursue a career in computer science. The U.S Bureau of Labor Statistics projects that jobs requiring a bachelor's or master's degree in computer science will grow 16 percent from 2018 to 2028 [2]. Unfortunately, the qualified computer scientists that universities are churning out does not even come close to meeting this demand. While 67 percent of all new STEM jobs created today involve computing, computer science degrees make up only 11 percent of STEM bachelor's degrees [3]. And apart from the economics of a computer science degree, the massive demand by parents and teachers for growing computer science in high schools is not being met. Nine out of ten parents want their child to be exposed to computer science in some form, but only about a quarter of high schools nationwide actually teach computer science [1], [3].

Too often is computer science postponed until the start of a student's undergraduate career. Even before critically examining the shortcomings of this approach, it is worth stepping back and considering the big picture. Would any student begin pursuing a bachelor's in mathematics without ever having taken a math class? What about English? This knee-jerk "of course not" is all too obvious here, but computer science still comes off as rather arcane to many educators, and the social norms and stigmas surrounding the field serve as a barrier to entry that actively disservices the next generation of thinkers, which may very well be rife with Turings and Knuths. And it is well-established that youth will more easily develop and rearrange neural synapses in the prefrontal cortex that are essential for the abstract causal reasoning so vital to the field [4].

It will be a waste of time and money if a stable computer science education system is built from the ground up, but is not taken advantage of. The sobering reality, as alluded to earlier, is that few students will actually continue with computer science until it develops into a career. Getting students excited about computer science, to the extent that they are willing to dedicate the rest of their lives to it, starts well before they even begin to think about college. If academic institutions do not foster this enthusiasm early on, it will likely never come to fruition [5]. It is thus a central belief of the authors of this paper that the most pressing concern of the future of computer science rests in the manner in which students are introduced to it.

Data for this project was sourced from a number of different outlets. The primary data source we relied on is Wisconsin's Department of Public Instruction. The DPI collects and maintains important information such as teacher licensure information, overall school performance, enrollment statistics, and educational finance data that is needed for local and state reporting [6]. We submitted requests to the DPI for access to confidential data that would aid in our research, such as course information and demographic breakdowns. By in large, these requests were accepted. However, constructing a more complete mosaic of the computer science landscape demands that we supplement this data with records from other sources.

Challenges

On the surface, this research may seem trivial, only re quiring a web browser and a few strokes on a keyboard to obtain all relevant information. However, the data available to us is anything but ideal. Records from the DPI are noisy, incomplete, outdated, and are often in direct conflict with other sources. This is a direct result of a lack of standards of any sort to govern how computer science education is reported to the state, as well as a general ignorance among many educators as to what constitutes computer science. A serious hurdle in this research was separating courses with genuine computer science content from those concerned with typing, business education, or learning to use Microsoft Excel. Furthermore, there is no mention of computer science in Wisconsin's academic state standards [7]. This lack of a unified voice behind computer science fundamentals makes the work of PUMP-CS all the more important.

Another issue that came up during the course of this project was scraping data from individual high school websites in an attempt to supplement the incomplete knowledge of the DPI. Unlike the DPI website, there is no standard for how information is displayed on school websites, thwarting an attempt at automating the process. We briefly considered a Python package called Beautiful Soup, used to search HTML and XML files by navigating the parse tree [8]. However, we quickly discovered that adequately refining search parameters for the script to function would require an amount of time that would ultimately defeat the purpose. So, this data was collected either by browsing the school's website, or calling the school's number to ask for information.

Analysis and Visualizations

In this research, we have encountered many dozens of courses that quality as computer science. However, a few of these courses stand out as ideal, and their successful implementation is often a testament to the overall health of a CS department. They are AP Computer Science A, AP Computer Science Principles, and Exploring Computer Science.

AP Computer Science A

AP Computer Science A is routinely viewed as the quintessential object-oriented programming course that prepares students to continue the study of computer science into college. Taught through the Java language, AP CS A is a problem-intensive course, with common exercises requiring students to design and develop solutions to abstractions of real-world problems [9]. In addition to coding solutions to problems, AP CS A challenges students to come up with the most efficient way of solving a problem, and to code the solution in a manner that another programmer can easily understand and build off of it. Paradigms of this course may be difficult to grasp for many students, and as such it is recommended that students have a strong foundation in algebra and mathematical reasoning. The quick pace of the course also makes it beneficial for students to have had some previous coding experience, whether OOP or functional. As such, this course may not be ideal for many high school students, thus necessitating the need for the other courses mentioned.

AP Computer Science Principles

AP Computer Science Principles is similar in structure to AP Computer Science A but is less technical and spends more time focused on the process of solving problems as well as the broader impact that computing has on the world [10]. Unlike AP CS A, AP CS Principles does not mandate Java, and instead instructors are given the freedom to decide what language the class should be centered around. AP CS Principles takes a more ”big picture” approach to the computer science universe, exploring such topics as parallel/distributed computing and fault tolerance. This course does not carry the same mathematics background recommendation that AP CS A does, and no previous coding experience is required. The course material is tailored to the student who may be curious about computer science and wants a new experience. Consequently, this makes AP CS Principles an excellent choice for schools looking to get more students interested in the field, as AP CS A students typically have this interest beforehand.

Exploring Computer Science

Exploring Computer Science was created specifically to target the socioeconomic disparities that plague computer science [11], and ultimately seeks to broaden participation in the field [12]. ECS is very similar to AP Computer Science Principles, but with some major paradigm shifts. Firstly, ECS places a strong emphasis on the integration of computer science into popular culture and also focuses on the creative elements of computing, resulting in CS becoming an art of sorts as opposed to just another math class. ECS largely does away with a lot of the technical jargon found in the other two courses as a way to reduce barriers to entry into the course. The course material is often less concerned with the ‘how’ of AP CS A and instead encourages students to examine the ‘why’ diving into the associated ethical and social concerns with one technology or another. Exploring Computer Science is the gold standard for making the field more equitable, and the results of implementing this course nationwide have shown major strides in shrinking gender and race-based disparities [13].

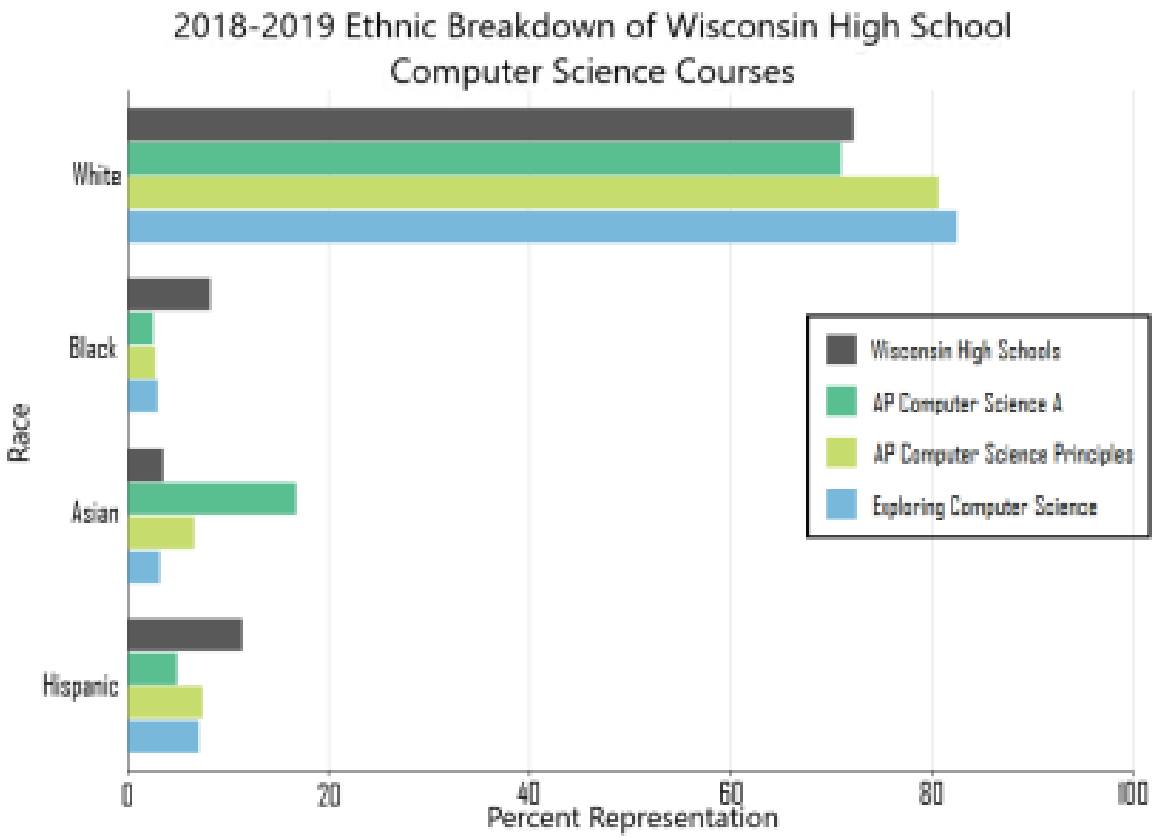

Fig. 1. Bar chart of computer science course and race data.

CS Enrollment

This research indicates that the number of students enrolled in a course with sufficiently high computer science content has greatly increased in recent years, even with overall decreasing high school enrollment. This growth is detailed in Figure 2, analyzing the three main courses discussed earlier. Although some growth is observed in AP Computer Science A, the vast majority of growth is seen in AP Computer Science Principles and Exploring Computer Science. Increasing enrollment in these two courses is very beneficial, as both are tailored in such a way to increase general access to computer science and are often offered in areas where students would not have another option. That being said, we have not observed the same degree of racial and economic diversification of computer science enrollment as much of the rest of the country has with these particular courses. Figure 1 shows that with respect to high school composition, white students are over-represented in 2 of the 3 courses and Asians in all 3. Black and Hispanic students, on the other hand, are underrepresented in all 3 courses. Furthermore, Figure 3 demonstrates that the median family income of a county is somewhat positively correlated with access to the AP CS A exam. However, it is difficult to draw a generalization from these observations as Wisconsin tends to be a mostly white state, and many other factors contribute to economic status.

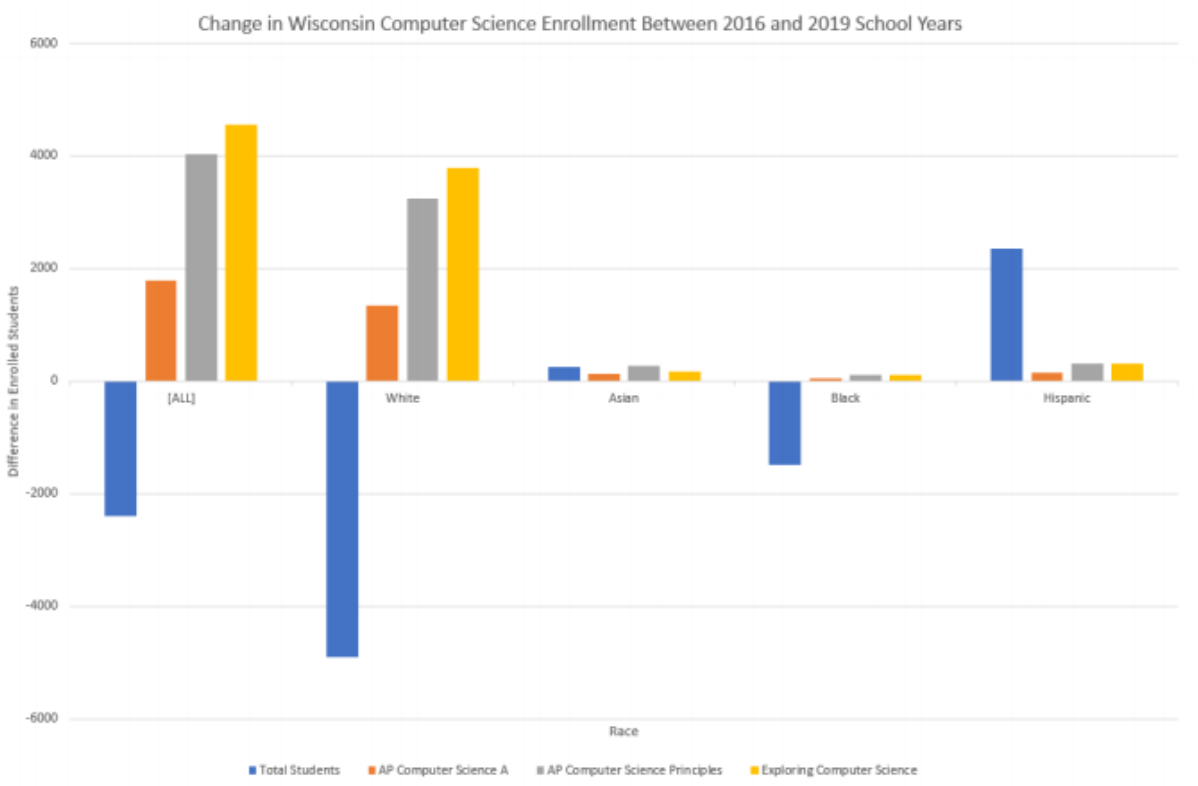

Fig. 2. Bar chart of computer science course and race data.

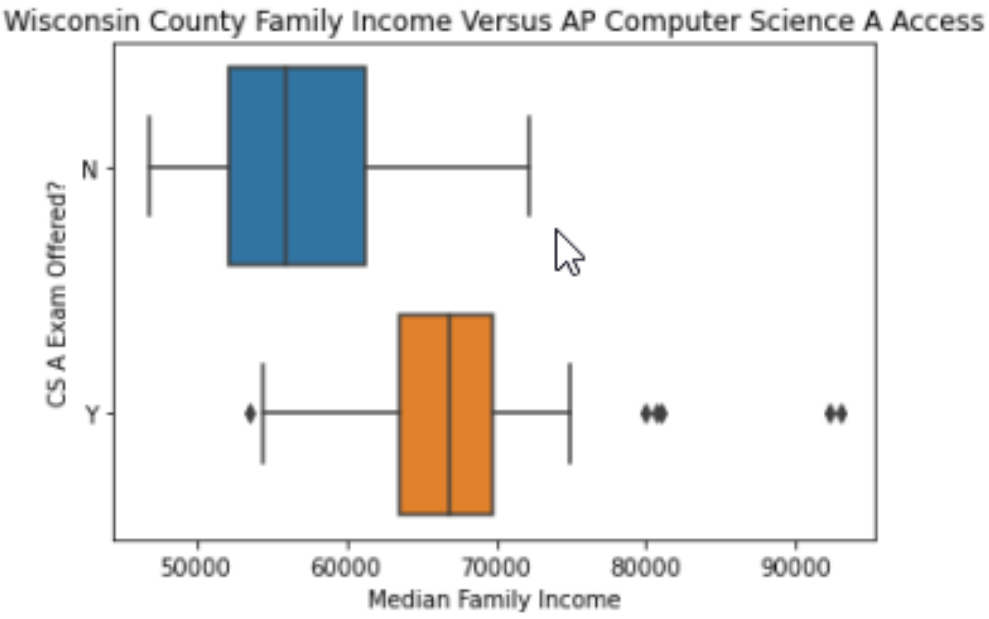

Fig. 3. Box plot of family income and access to AP CS A exam

CS Instruction

One serious caveat that exists in measuring course enrollment is also tracking the available resources to sustain growing enrollment. Unfortunately, the results of this research

suggest that there is not a sufficient number of teachers with computer science licenses in the state of Wisconsin to support the rapidly expanding field. Roughly 50% of the high schools that teach computer science in the state do not have a single teacher with a computer science license. Figure 5 shows the with/without licensed CS teacher breakdown for a handful of courses.

2018-19 CS High School Course Density

Course Name	With licensed teacher at school	Without licensed teacher at school
AP Computer Science A	32	35
AP Computer Science Principles	19	23
Exploring Computer Science	12	37
Computer Programming	35	36
Mobile Applications	6	11
Computer Game Design	18	45

Fig. 5. Bar chart of computer science course and race data.

References

[1] Google Inc. & GALLUP Inc., Trends in the State of Computer Science in U.S. K-12 Schools. Retrieved from http://goo.gl/j291E0

[2] Bureau of Labor Statistics, U.S. Department of Labor, Occupational Outlook Handbook. https://www.bls.gov/ooh/computer-and-informationtechnology/computer-and-information-research-scientists.htm

[3] Code.org, Summary of source data for Code.org infographics and stats https://docs.google.com/document/d/1gySkItxiJn vwb8HIIKNXqen184m RtzDX12cux0ZgZk/pub

[4] A. Gopnik, T. Griffiths, and C. Lucas, "When Younger Learners Can Be Better (or at Least More Open-Minded) Than Older Ones." https://cocosci.princeton.edu/tom/papers/LabPublications/GopnicketalYo ungLearners.pdf

[5] Google Inc. & GALLUP Inc., Encouraging Students Toward Computer Science Learning. https://goo.gl/iM5g3A

[6] Wisconsin Department of Public Instruction, https://dpi.wi.gov/wise/datarequests

[7] Wisconsin Department of Public Instruction, WISCONSIN STATE STANDARDS for Literacy in All Subjects. https://dpi.wi.gov/standards/guidingprinciples

[8] Leonard Richardson, "Beautiful Soup 4.9.0 Documentation." https://www.crummy.com/software/BeautifulSoup/bs4/doc/

[9] College Board, "Advanced Placement Computer Science A." https://apstudents.collegeboard.org/courses/ap-computer-science-a

[10] College Board, "Advanced Placement Computer Science Principles." https://apcentral.collegeboard.org/courses/ap-computer-science-principles

[11] Margolis, Jane. (2008). Stuck in the shallow end : education, race, and computing. Cambridge, Mass. :MIT Press https://ieeexplore.ieee.org/book/6267411

[12] Exploring Computer Science, "What is ECS?" http://www.exploringcs.org/

[13] Exploring Computer Science, "Student Indictators Research." http://www.exploringcs.org/for-researchers-policymakers/reports/results