Paper Title Expanding Computer Science for All: Effectiveness of State Practices and Implementations

Author(s) Victoria Concepción Chávez, Northwestern University School of Education and Social Policy Learning Sciences

Session Title Implementation of STEM and Technical Education Policies

Session Type Roundtable Presentation

Presentation Date 4/11/2021

Presentation Location Virtual

Descriptors Career and Technology Education, Case Studies, Technology

Methodology Conceptual/Theoretical

Unit Division L - Educational Policies and Politics

DOI https://doi.org/10.3102/1688247

Expanding Computer Science for All:
Effectiveness of State Practices and Implementations

Victoria C. Chávez
University of Rhode Island

Abstract

In the last few years, there has been a strong push toward getting computer science (CS) into K-12 classrooms. Namely, many CS education activists are trying to get computer science into all public schools across the nation. Utilizing institutional theory (IT) to analyze educational policy and practice, this case study examines how and why certain state-level implementations have been more effective than others in expanding access to K-12 CS education. In line with Sharan Merriam's definition of a case study, we begin with a brief literature review. Semi-structured interviews with key actors from CS education coalitions from California, Illinois, and Rhode Island provide further insight into concrete implementation differences and their effects on increasing access to K-12 CS education.

Expanding Computer Science for All:
Effectiveness of State Practices and Implementations

Objective

In the last few years, there has been a strong push toward getting computer science (CS) into K-12 classrooms. Namely, many CS education activists are trying to get computer science into all public schools across the nation. The movement started in the mid-2010s and was supported by several cities, states, President Obama in 2016 (White House Archives, 2016), and continued to be supported under the previous White House administration (White House Archives, 2017).

At a local level, some states have embraced the movement more than others. Implementations vary wildly, ranging from enacting policy decisions about CS education-- such as having board-approved strategic plans or CS standards-- to simply having publicly stated support from local politicians. The purpose of this case study is to examine the various levels of support CS education has across several states and to determine what makes some implementations more effective than others in terms of increasing access to CS education.

Theoretical framework

Drawing from Patricia Burch's (2007) work on utilizing institutional theory (IT) to analyze educational policy and practice, this paper examines how and why certain implementations have been more effective than others in expanding access to K-12 CS education. Beyond using the core constructs of IT to understand why CS has been a growing need and a driving force in education over the last decade, an analysis of the organizational fields: local (state-level) CS education advocacy groups, and the field outcomes: increased access to CS education as measured by percentage of local high schools offering CS courses and AP exams taken, will allow us to gain a better understanding of what and how field effects lead to better outcomes.

Methods and Data

In line with Sharan Merriam's (2001) definition of a case study, to fully study the phenomena of "computer science for all," we first conduct an analysis on the literature surrounding CS education. This case study consists of a brief literature review analyzing: the definitions and differences of computer science vs. computational thinking, the benefits of learning CS, and the current progress of getting CS into K-12 classrooms across the United States. Lastly, semi-structured interviews with key actors from CS education coalitions from California, Illinois, and Rhode Island provide further insight into concrete implementation differences and their effects on increasing access to K-12 CS education. Given the limited data available regarding K-12 computer science courses, 'increased access to CS education' is measured by percentage of local high schools offering CS courses and AP CS exams taken, as reported by Code.org and The College Board.

Literature Review

Definitions

In its simplest form, CS is about computation or calculation (Wing, 2006); in one of its most extensive definitions, it is "the study of computers and algorithmic processes, including their principles, their hardware and software designs, their applications, and their impact on society" (Tucker et al., 2003). Computer science is a tool through which to learn and teach computational

thinking (CT) skills, which involve "solving problems, designing systems, and understanding human behavior" (Wing, 2006). Computational thinking is associated with creativity, critical thinking, and problem solving (Ananiadou and Claro, 2009; Binkley et al., 2012) and as such, teaching CS entails more than simply teaching students how to code.

While one can learn how to code without learning how to solve complex problems, computational thinking entails thinking systematically about problems and their solutions—one does not need to learn how to program in order to learn how to think computationally about problems. Note that Tucker et al.'s (2003) definition of computer science, "the study of computers and algorithmic processes, including their principles, their hardware and software designs, their applications, and their impact on society," does not even explicitly mention programming. In order for students to be able to become computational thinkers and "creators of technology and not just users of it" as Julie Flapan (personal communication, December 5, 2018), Executive Director of Alliance for California Computing Education for Students and Schools, put it, they must understand algorithmic processes and computing principles and they must be able to apply said concepts outside of the context of programming. In fact, there have been several 'unplugged' CS curricula developed—curricula that require no technology to be implemented— that still teach computer science fundamentals and that still help develop computational and critical thinking as well as problem-solving skills.

Benefits of Learning Computer Science

An important aspect of computer science, and a crucial piece to ensuring students are educated users and consumers of technology, is understanding the impact technology has on society. With a continually growing online presence, constant scandals of data and security breaches, and rising questions around the ethics of topics such as machine learning and artificial intelligence, students must be able to understand where and how technology is affecting their lives. Having a basic understanding of how these technologies work is the first step in being able to make informed decisions about the products they use and how they use them.

Another argument for teaching computer science is to not only provide students with the digital literacy skills needed to perform everyday tasks in the 21st century and be educated consumers of technology, but to also help them develop computational and critical thinking as well as problem-solving skills (Code.org & Computing in the Core). While there is research in favor of the computational thinking benefits associated with learning computer science, there is little to no research on learning CS and related student outcomes on core subjects or even CS itself. If computer science does indeed teach students computational thinking skills, should this not be reflected in, say, higher test scores on reading, math, and science? The biggest research endeavor around these types of questions is currently underway in New York through an extensive 9-year evaluation of New York City's CS education rollout through CS4All NYC (Villavicencio, Fancsali, and Martin, 2018). In the meantime, researchers have found that CS can help deepen students' understanding of mathematics (Papert, 1972), foster and advance computational thinking skills (Pea and Kurland, 1984; Lye and Koh, 2014; Atmatzidou and Demetriadis, 2016), and increase student interest in pursuing computer science as well as other scientific fields (Dettori, Greenberg, McGee, and Reed, 2016).

In a paper from 1984, Roy D. Pea and D. Midian Kurland conduct an extensive literature review to assess the theory about whether or not computer programming is a good tool for helping students develop problem solving skills. While they do not find conclusive evidence in favor of the theory, they do find that it is possible to design learning experiences "[wherein] the course of

learning to program students are confronted with new ideas and have opportunities to build them into their own understanding of the computer system and computational concepts." Namely, a 1972 paper argued that using Logo turtle to write programs allowed students to better understand geometric concepts that would otherwise be much more difficult to explain (Pea and Kurland, 1984). While Papert conducted no experiments and did not document any background for his observations, he claims watching students make "spontaneous discoveries" regarding abstract geometric concepts, through their use of programming the turtle to visualize said concepts.

To conclude their literature review, Pea and Kurland explore the concept of knowledge transferability—does the process of learning how to program transfer to other contexts? They conclude:

> As reading is often equated with skill in decoding, "learning to program" in schools is often equated with learning the vocabulary and syntax of a programming language. But skilled programming, like reading, is complex and context-dependent, so we must begin to unpack the contexts in which programming is carried out and learned (1984).

They go on to describe the importance of a programming environment and argue that learning how to program in a good programming environment allows students to be able to focus "on higher level issues of program design, efficiency, and elegance." The design, efficiency, and elegance they refer to, are what others would now refer to as computing fundamentals—the precise fundamentals that allow students to develop critical thinking skills.

An updated literature review from 2014 finds that indeed programming can help foster computational thinking skills. For the purpose of that paper, the researchers define programming very differently from coding; in essence, they define it as learning computer science fundamentals: "programming is more than just coding, for, it exposes students to computational thinking which involves problem-solving using computer science concepts like abstraction and decomposition" (Lye and Koh, 2014). Lye and Koh analyze seven studies conducted at the K-12 level, all of them showing correlations between learning computer science and positive effects on students' computational thinking skills. They find research suggesting that visual programming languages, such as Scratch, Alice, Blockly, and the like, "can potentially allow students to acquire the computational concepts more easily without the need to learn complex programming syntax" (2014). They conclude that the visual representation of these programming languages allows students to not only visually relate computational concepts but also allow students to lift a lighter cognitive load when it comes to issues of testing and debugging, allowing them to "acquire computational problem-solving practices more easily" (Lye and Koh, 2014). While the studies analyzed are mostly qualitative, they shed light on some possible positive effects of learning computer science and should encourage more conclusive research efforts.

In a more recent and exhaustive study, researchers find that educational robotics help advance students' computational skills. In the Greek study consisting of 164 students aged 15 and 18, researchers measure the students' CT skills before, during, and after a robotics activity. They find that although there are differences in how long it takes students of different ages and genders to reach the same computational thinking skill level, all students eventually reach the same level, with most of the improvement appearing toward the end of the activity (Atmatzidou and Demetriadis, 2016). While it is unclear how exactly CT skills are measured, the use of a quantitative metric before, during, and after the robotics activity allows researchers to see the direct effects of the activity. Another study from 2016 shows a correlation between learning computer science in a culturally responsive manner and becoming more interested in pursuing both computer science as well as other scientific fields. The qualitative study based out of public Chicago high

schools uses pre and post student surveys to gauge student interest in computing and other scientific fields. It finds that 74% of students enrolled in a course using the Exploring Computer Science curriculum showed "increased interest in taking another computer science course or majoring in computer science in college" and somewhat of an increased interest in majoring in science, engineering, and mathematics (Dettori, Greenberg, McGee, and Reed, 2016).

K-12 CS: State of Affairs

In order to help states expand CS as quickly as possible, Code.org, a national non-profit known for its advocacy for computer science education, outlined nine policy recommendations for expanding CS education (Code.org, 2019). Between 2013 and 2018, the number of states to have implemented at least one of the nine policies has more than tripled and as of 2019, all 50 states have adopted or are in the process of adopting at least one of the nine policies (Code.org, 2019). According to Code.org's 2019 *State of Computer Science Education Report*, across the 39 states analyzed, 45% of high schools teach CS. Furthermore, Rhode Island and Arkansas are among the states to have made policy decisions regarding CS education and have the greatest percentage of CS being taught in schools.

Unfortunately, the success of implementing computer science in K-12 public schools varies greatly from state to state (for a complete per-state and per-policy breakdown, refer to Figure 1 and Table 1 in the Appendix) and the whole picture of CS education is not as promising as one might think. Firstly, the two states mentioned above stand out from the other three dozen analyzed, precisely because they have had such aggressive policy implementations that have allowed them to expand access to computer science quickly. More than 60% of states have less than half of their public high schools teaching CS, with Alaska, Louisiana, and New Mexico all having less than 25% of their high schools offering CS.

Analysis and Findings

Looking at CS Education Advocacy Groups in Three States

With a more nuanced understanding of the motivations for and progress of expanding computer science education in K-12, one can now analyze how organizational fields' approaches to expanding K-12 CS education have contributed to measurable field outcomes. Given DiMaggio's and Powell's (1983) definition of organizational fields: "organizations that in the aggregate constitute a recognized area of institutional life: key suppliers, resource product consumers, regulatory agencies and other organizations that produce similar products or service," the organizations analyzed below are the statewide CS education advocacy groups of three states in particular: Rhode Island, California, and Illinois. In line with Burch's (2007) articulation of field effects, "increases in the ideological, human, and financial resources available to individuals or organizations working on an issue," the field effects of K-12 CS education policy implementation, funding, and awareness can all contribute to the outcomes being measured, in this case: percentage of local high schools offering CS courses and AP CS exams taken.

Rhode Island

Rhode Island's CS education advocacy group, CS4RI, "is a state initiative to bring high quality computer science (CS) learning experiences to all students. It represents a partnership between Rhode Island state government, the Rhode Island Department of Education, K-12 schools, higher education, private industry, and non-profits across Rhode Island" (CS4RI, n.d. -a). With support and funding from the governor and the state's department of education, CS4RI partnered

up with several program providers who offered different levels of computer science curricula, ranging from modular units, to a combination of algebra and CS, to standalone computer science courses (CS4RI, n.d. -b). The variety of curricula allow schools and teachers to select a program that meets their needs and available resources. To help ensure students are meeting the same learning objectives, the Rhode Island Department of Education passed CS standards in the spring of 2018. The standards are divided into subject areas and CS pathways were also developed so schools can implement rigorous and cohesive sequenced courses. The group is currently working on micro-credentialing for CS teachers, as well as working on expanding work-based learning opportunities for high school students (CS4RI, n.d. -c).

The state has implemented several of Code.org policies, among them: creating a state implementation plan, developing K-12 CS standards, funding teacher professional learning, and allowing CS to count towards a graduation requirement (Code.org, 2019). According to their website and former RI Governor Gina Raimondo, as of December 2017, every public school n the state taught computer science (CS4RI, n.d. -b). However, the definition of 'teaching computer science' is quite loose and ranges from schools having had taught 'An Hour of Code' to schools having full CS pathways that include students receiving college credit.

California

Compared to CS4RI, ACCESS (Alliance for California Computing Education for Students and Schools) has significantly less political support and funding from the state and board of education. In speaking with Julie Flapan, Executive Director of ACCESS, she mentioned ACCESS's goal of all high schools offering computer science by 2020 and all students having access to it by 2025. By 2025, the goal is also that both participation and passing rates will mirror the population of California. In 2019, the state approved an implementation plan and adopted CS K-12 standards (Code.org, 2019). The state has also adopted CS teacher credentialing and supports-- but does not require— districts to allow for CS to count towards a graduation requirement.

While California has slowly increased access to CS, they are still far from their goal. In 2016, only 0.5% of 1.9 million high school students in the state took the AP CS A exam (ACCESS and Kapor Center for Social Impact) and as of 2019, only 47% of their high schools taught CS. During her interview, Flapan emphasized the importance of paying attention to access and equity, even if it meant taking longer to reach every school (personal communication, December 5, 2018). Of the students who took an AP exam in 2017, 30% were female and 27% were underrepresented minorities—higher than the national rates of 27% and 20%, respectively, during the same year (Code.org, 2019; Chang, 2017).

Illinois

Unlike Rhode Island or California, Illinois' CS4IL program is much newer and, for most of its existence, was completely self-funded by a single passionate CS teacher. Steven Svetlik, long-time CS teacher, saw the need for a CS for Illinois organization in 2017 so he took the initiative to begin one. The group, approved as a 501(c)(3) organization in 2019, consists of about 40 members, most of whom are all passionate CS teachers (personal communication, December 7, 2018).

In May 2018 there was a House Resolution adopted that "encouraged" Illinois school districts to teach "computer science and coding" (HR0853, 2018). However, the state of Illinois still has no legislation or financial support for CS education. Standards have not been developed

and the state does not yet have "an operational definition of K–12 computer science education," despite it being one of the recommendations given in 2017 by the Illinois Task Force on Computer Science Education (personal communication, December 7, 2018; Illinois Task Force on Computer Science Education, 2017). Svetlik raised concerns about CS education being inaccessible to rural communities in the state, as a lot of the efforts and successes of CS education have been primarily focused on the city of Chicago and surrounding suburbs. As of 2019, only 37% of Illinois's high schools teach computer science (Code.org, 2019).

Analysis

Of the three states analyzed, Rhode Island has by far the most and the most, and arguably strongest, field effects: support from the Governor, funding from government agencies and local industry partners, access to financial and human resources at the RI Department of Education, and a strong established presence as an organization in the state. It also has the highest field outcomes as measured by percentage of local high schools offering CS courses and AP CS exams taken. Per Code.org's report, the state of Rhode Island (RI) has one of the most successful implementations of computer science in the nation, offering CS in 86% of its high schools (Code.org, 2019). According to The College Board (2017, 2020), not only have the number of AP CS exams taken between 2017 and 2020 increased by about 12% and 201% for AP CS A and AP CS Principles, respectively, but the average scores for AP CS A have also increased by about 14%. While these trends are common across the board, Rhode Island has seen higher increases in exams taken and average scores than California, Illinois, and the nation as a whole (see Table 2).

As outlined in Table 2 in the Appendix, Illinois and California have also seen field outcomes relative to their field effects. While both states have had some field effects contributing to expanded CS education, both states have struggled to receive systemic government-level support with Illinois being farther behind when compared to California and Rhode Island. Both Illinois and California show increases in exams taken on par with national ones but while Illinois' average scores of both AP CS exams have decreased between 2017 and 2020, California's average scores of both AP CS exams have increased. California's increase in average AP CS Principles scores is particularly notable as both other states and nationwide scores have decreased over the last four years. Lastly, the percentage of high schools offering at least one computer science course for California is around the national average—47% vs 45%-- while Illinois trails behind at 37%.

Significance

Technology is evermore present in everything that we do as a society and it is becoming increasingly necessary to have basic computer skills in order to perform everyday tasks. However, it is estimated that 16% of Americans between the ages of 16 and 65 are not digitally literate (U.S. DOE, 2018). Furthermore, while there are currently half a million computing jobs available, there were less than 75,000 computer science college graduates in 2018 (Code.org, 2020). Given this pressure and the need for people with more advanced computer and computational skills, it is only reasonable to look to the education system to fill this need, yet much work remains to be done in this area, especially with regard to research and best practices.

Code.org has released annual reports outlining the progress of expanding CS education in K-12 across the nation since 2017. While their reports have highlighted the correlation between states' implementation of Code.org's nine policies and increased AP CS exams taken as well as increased percentage of high schools offering computer science courses, no clear research has been done on which of these policies are most effective and in what contexts. Using institutional theory

to analyze states' approaches to expanding computer science education gives an opportunity to better understand the effects of each policy and in turn allows states to better allocate limited resources.

Conclusion and Discussion

The lack of research available around CS education and student outcomes may be cause for some states being hesitant to provide the necessary support and funding for CS education. Similarly, a lack of research around best mass-scale implementation practices leaves states and organizations without guidance as to how to implement practices and policies such as those set forth by Code.org. It is important for more research to be done to iterate and improve early on in the initiatives' lifetimes and to understand the effects of expanding CS education. Future work should also include analyzing the impacts of individual Code.org policies and the effects of implementation fidelity on increasing access to CS education (see Appendix).

Lastly, it is important to assess and address issues of equity, as CS has the potential to either exasperate school inequities or begin evening the playing field for students. It is important to ensure that the programs being scaled are high-quality and that they do not reproduce the inequities already present in the education system but instead work to disrupt it-- race, gender, socio-economic status, disability, and urbanicity should not be determinants of students' access to quality CS education. As best put by Flapan, states should not be so concerned with scaling up "so fast and furiously that [they] fail to attend to the quality and equity of what [they are] scaling" (personal communication, December 5, 2018).

References

ACCESS-InNeedofRepair.pdf. (n.d.). Retrieved from http://access-ca.org/wp-content/uploads/sites/4/2015/08/ACCESS-InNeedofRepair.pdf

Alliance For California Computing Education For Students And Schools, & Kapor Center for Social Impact. (n.d.). *KC16017_Report-Final.pdf*. Retrieved from http://access-ca.org/wp-content/uploads/sites/4/2017/01/KC16017_Report-Final.pdf

Ananiadou, K. and Claro, M. (2009). *21st Century Skills And Competences For New Millennium Learners In OECD Countries*. OECD Publishing

Atmatzidou, S., & Demetriadis, S. (2016). Advancing students' computational thinking skills through educational robotics: A study on age and gender relevant differences. *Robotics and Autonomous Systems*, *75*, 661–670. https://doi.org/10.1016/j.robot.2015.10.008

Bernier, D., & Margolis, J. (2014, July 2). The Revolving Door: Computer Science for All and the Challenge of Teacher Retention. Retrieved from http://www.exploringcs.org/wp-content/uploads/2014/04/The-Revolving-Door-CS-for-All-and-the-Challenge-of-Teacher-Retention-Final.pdf

Binkley, M., Erstad, O., Herman, J., Raizen, S., Ripley, M., Miller-Ricci, M., and Rumble, M. (2012). Defining twenty-first century skills. In *Assessment And Teaching Of 21st Century Skills*, pages 17–66. Springer.

Burch, P. (2007). Educational Policy and Practice from the Perspective of Institutional Theory: Crafting a Wider Lens. *Educational Researcher, 36*(2), 84-95. Retrieved from www.jstor.org/stable/4621078.

Burgstahler, S. (2011). Universal Design: Implications for Computing Education. *ACM Transactions on Computing Education*, *11*(3), 1–17. https://doi.org/10.1145/2037276.2037283.

Calabrese Barton, A., & Tan, E. (2010). We Be Burnin'! Agency, Identity, and Science Learning. *Journal of The Learning Sciences - J LEARN SCI*, *19*, 187–229. https://doi.org/10.1080/10508400903530044.

Chang, R. (2017). 29,000 Female Students Took AP CS Exam in 2017. *The Journal: Transforming Education Through Technology*. Retrieved from http://thejournal.com/articles/2017/07/19/29000-female-students-took-ap-cs-exam-in-2017.aspx.

Code.org, & Computing in the Core. (n.d.). What is Computer Science and What do People Do Once They Know It? Retrieved from https://code.org/files/computer_science_is_foundational.pdf

Code.org (2018). *2018 State of Computer Science Education: Policy and Implementation*. Retrieved from https://code.org/files/2018_state_of_cs.pdf.

Code.org. (2019). *2019 State of Computer Science Education: Equity and Diversity.* Retrieved from https://advocacy.code.org/2019_state_of_cs.pdf.

Code.org. (2020). Why Computer Science?. Retrieved from https://code.org/promote.

College Board, The (2017). National Summary Report. AP Data – Archived Data 2017. Retrieved from https://research.collegeboard.org/programs/ap/data/archived/ap-2017

College Board, The (2020). National Summary Report. AP Program Participation and Performance Data 2020. Retrieved from https://research.collegeboard.org/programs/ap/data/participation/ap-2020

Computational thinking in compulsory education: Towards an agenda for research and practice | SpringerLink. (n.d.). Retrieved November 19, 2018, from https://link.springer.com/article/10.1007/s10639-015-9412-6

CS4RI. (n.d. -a). CS4RI-- A State-Wide Initiative. CS4RI. Retrieved from https://www.cs4ri.org/what-we-do.

CS4RI. (n.d. -b). CS4RI for School Teachers & Administrators. CS4RI. Retrieved from https://www.cs4ri.org/overview.

CS4RI. (n.d. -c). News & Events. CS4RI. Retrieved from https://www.cs4ri.org/new-events.

Cuban, L. (n.d.). *Oversold and Underused : Computers in the Classroom*. Retrieved from https://ebookcentral-proquest-com.revproxy.brown.edu/lib/brown/reader.action?docID=3300336&query=

Cuny, J. (n.d.). Infosys Foundation USA - Media | Blog | Computer Science Education for Everyone: A Groundswell of Support. Retrieved November 18, 2018, from http://www.infosys.org/infosys-foundation-usa/media/blog/Pages/groundswell-support.aspx

Dettori, L., Greenberg, R. I., McGee, S., & Reed, D. (2016). The Impact of the Exploring Computer Science Instructional Model in Chicago Public Schools. *Computing in Science Engineering*, *18*(2), 10–17. https://doi.org/10.1109/MCSE.2016.39

DiMaggio, P. J., & Powell, W. W. (1983). The iron cage revisited: Institutional isomorphism and collective rationality in organizational fields. *American Sociological Review*, 48, 63-82.

Escueta, Maya et al. (2017). "Education Technology: An Evidence-Based Review," National Bureau of Economic Research Working Paper 23744, 1-89.

Evaluating the Reach, Quality, and Impact of Computer Science for All in NYC - Projects – NYU Steinhardt. (n.d.). Retrieved November 17, 2018, from https://steinhardt.nyu.edu/research_alliance/research/projects/CS4All

Gallup. (2016). *Trends in the State of Computer Science in U.S. K-12 Schools*. Retrieved from http://services.google.com/fh/files/misc/trends-in-the-state-of-computer-science-report.pdf

Goode, J. (n.d.). Connecting K-16 Curriculum & Policy: Making Computer Science Engaging, Accessible, and Hospitable for Underrepresented Students, 5.

Goode, J., & Margolis, J. (2011). Exploring Computer Science: A Case Study of School Reform. *ACM Transactions on Computing Education*, *11*(2), 1–16. https://doi.org/10.1145/1993069.1993076

Guzdial, M. (2015). Learner-Centered Design of Computing Education: Research on Computing for Everyone. *Synthesis Lectures on Human-Centered Informatics*, *8*(6), 1–165. https://doi.org/10.2200/S00684ED1V01Y201511HCI033

Landscape of CS Action in States. (n.d.). Retrieved November 18, 2018, from https://docs.google.com/document/d/1J3TbEQt3SmIWuha7ooBPvlWpiK-pNVIV5uuQEzNzdkE/edit?usp=embed_facebook

Leskin, P., & Nov 12, 2018. (n.d.). Students in Brooklyn protest their school's use of a Zuckerberg-backed online curriculum that Facebook engineers helped build. Retrieved from http://www.businessinsider.com/summit-learning-school-curriculum-funded-by-zuckerberg-faces-backlash-brooklyn-2018-11

Lye, S. Y., & Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12? *Computers in Human Behavior*, *41*, 51–61. https://doi.org/10.1016/j.chb.2014.09.012

Margolis, J., Chapman, G., Goode, J., Dettori, L., & Lewis, D. (n.d.). A Tale of Three ECS Partnerships and Why Scalability ≠ Sustainability. Retrieved November 19, 2018, from http://www.exploringcs.org/wp-content/uploads/2014/04/A-Tale-of-Three-ECS-Partnerships.pdf

Margolis, J., Estrella, R., Goode, J., Holme, J. J., & Nao, K. (2010). *Stuck in the Shallow End: Education, Race, and Computing*. Cambridge, UNITED STATES: MIT Press. Retrieved from http://ebookcentral.proquest.com/lib/brown/detail.action?docID=3338925

Margolis, J., Ryoo, J. J., Sandoval, C. D. M., Lee, C., Goode, J., & Chapman, G. (2012). Beyond access: broadening participation in high school computer science. *ACM Inroads*, *3*(4), 72. https://doi.org/10.1145/2381083.2381102

Means, B., SRI International, M. P., CA, Education Development Center, I., & Newton, M. (1993). *Using Technology To Support Education Reform*. Washington, D.C.: Distributed by ERIC Clearinghouse.

Means et al. - 1993 - Using Technology To Support Education Reform.pdf. (n.d.). Retrieved from https://files.eric.ed.gov/fulltext/ED364220.pdf

Merriam, S. B. (2001) *Qualitative Research and Case Study Applications in Education*. San Francisco, CA: Jossey-Bass Publishers.

Miltner, K. M. (2017, December 8). Who Benefits From the Push to Teach Every Kid to Code? Retrieved from https://slate.com/technology/2017/12/who-benefits-from-the-push-to-teach-all-kids-to-code.html

Murphy, J. (2017). *Four Essential Skills For Workplace Success In The 21st Century Digital Economy.* Forbes. Retrieved from https://www.forbes.com/sites/forbescommunicationscouncil/2018/04/27/four-essential-skills-for-workplace-success-in-the-21st-century-digital-economy

Papert S. Teaching children to be mathematicians versus teaching about mathematics. International Journal for Mathematical Education, Science and Technology 3. 249 - 262 (1972b).

Pea, R. D., & Kurland, D. M. (1984). On the cognitive effects of learning computer programming. *New Ideas in Psychology*, 137–168.

Ryoo, J., Goode, J., & Margolis, J. (2015). It takes a village: supporting inquiry- and equity-oriented computer science pedagogy through a professional learning community. *Computer Science Education*, *25*(4), 351–370. https://doi.org/10.1080/08993408.2015.1130952

Ryoo, J. J., Margolis, J., Lee, C. H., Sandoval, C. D. M., & Goode, J. (2013). Democratizing computer science knowledge: transforming the face of computer science through public high school education. *Learning, Media and Technology*, *38*(2), 161–181. https://doi.org/10.1080/17439884.2013.756514

Sengupta, P. (n.d.). Integrating computational thinking with K-12 science education using agent-based computation: A theoretical framework | SpringerLink. Retrieved November 19, 2018, from https://link.springer.com/article/10.1007/s10639-012-9240-x

Sims, C. (Ed.). (2017). Front Matter. In *Disruptive Fixation* (pp. i–viii). Princeton University Press. Retrieved from https://www.jstor.org/stable/j.ctt1vwmh3c.1

Smith, M. (2016, January 30). Computer Science For All. Retrieved October 18, 2018, from https://obamawhitehouse.archives.gov/blog/2016/01/30/computer-science-all

Tucker, A. (2003). A Model Curriculum for K–12 Computer Science: Final Report of the ACM

K–12 Task Force Curriculum Committee. *The Association for Computing Machinery*.

Villavicencio, A., Fancsali, C., & Martin, W. (2018). Computer Science in New York City:, 82.

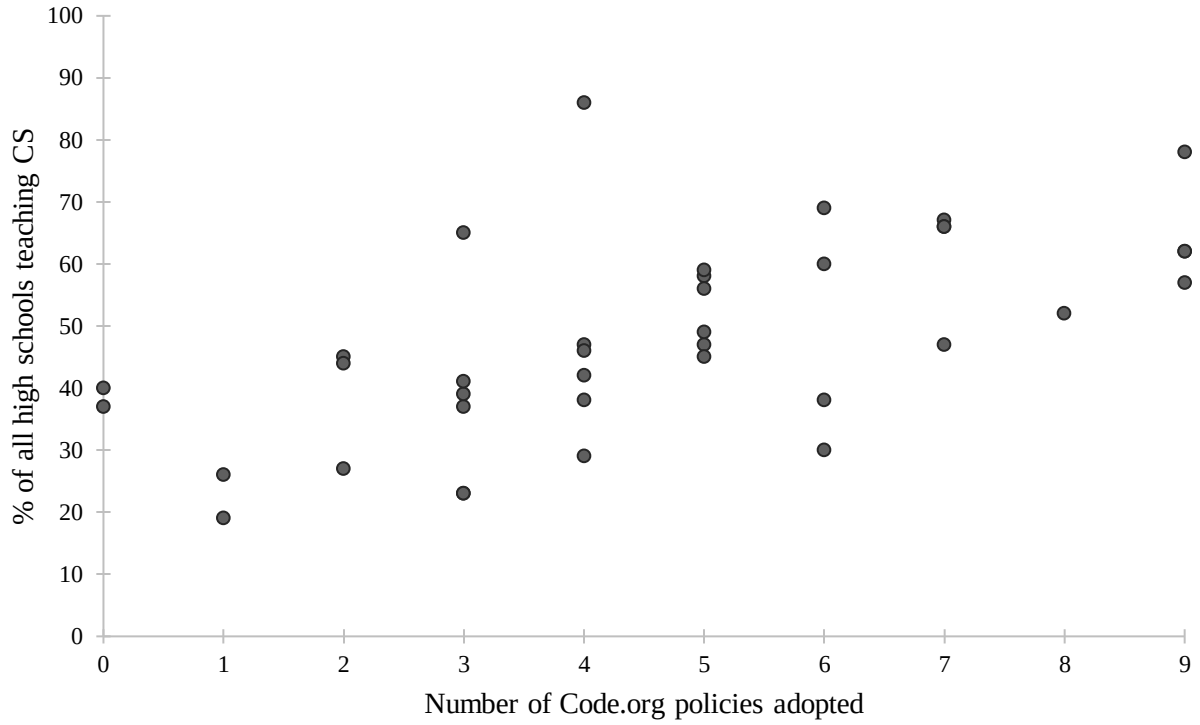
White House Archives, The. (2016). FACT SHEET: President Obama Announces Computer Science For All Initiative. *The White House*. Retrieved from https://obamawhitehouse.archives.gov/the-press-office/2016/01/30/fact-sheet-president-obama-announces-computer-science-all-initiative-0.

White House Archives, The. (2017). President Trump Signs Presidential Memo to Increase Access to STEM and Computer Science Education. *The White House*. Retrieved from https://trumpwhitehouse.archives.gov/articles/president-trump-signs-memorandum-stem-education-funding/.

Wing, J. M. (2006). Computational Thinking. Communications of the ACM, 49(3).
Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, *366*(1881), 3717–3725. https://doi.org/10.1098/rsta.2008.0118

Appendix

Figure 1: Relationship Between Code.org Policy Adoption and Percentage of High Schools Offering Computer Science, Per State*



*Data derived from Code.org's 2019 State of Computer Science Report. "Offering Computer Science" refers to schools offering at least one CS course during the 2017-18 or 2018-19 school year.

Table 1: Relationship Between Individual Policy Adoption and Average State Percentage of High Schools Offering Computer Science*

Code.org Policy	Average Percentage of H.S. Offering CS for States who Adopted Policy
Create a state plan for K-12 computer science	18.71%
Define computer science and establish rigorous K-12 computer science standards	37.16%
Allocate funding for rigorous computer science teacher professional learning and course support	30.26%
Implement clear certification pathways for computer science teachers	38.68%
Create programs at institutions of higher education to offer computer science to preservice teachers	19.42%
Establish dedicated computer science positions in State and Local Education Agencies	23.26%
Require that all secondary schools offer computer science with appropriate implementation timelines	22.11%
Allow computer science to satisfy a core graduation requirement	36.61%
Allow computer science to satisfy an admission requirement at institutions of higher education	19.42%

*Data derived from Code.org's 2019 State of Computer Science Report. "Offering Computer Science" refers to schools offering at least one CS course during the 2017-18 or 2018-19 school year.

Table 2: Relationship Between Field Effects and Field Outcomes as measured by AP CS exams taken, average AP CS exam scores, and percentage of local high schools offering at least one CS course.

Location	**Field Effects**	**% High Schools Offering At Least One CS Course (2019)**	**% Change AP CS A Exams Taken (2017-2020)**	**% Change AP CS A Mean Score (2017-2020)**	**% Change AP CS Principles Exams Taken (2017-2020)**	**% Change AP CS Principles Mean Score (2017-2020)**
IL	- Private funding	37%	12.82%	-0.89%	173.06%	-0.31%
CA	- CS standards - Teacher credentialing	47%	16.81%	3.92%	136.79%	1.92%
RI	- Government support (vocal governor, approved CS standards, early work on micro-credentialing) - Government funding - Private funding	86%	12.33%	14.56%	201.5%	-5.26%
National (all 50 states)	- Varies by state	45%*	15.89%	3.51%	160.82%	-2.53%

* Percentage based on only 39 states analyzed by Code.org in 2019.