“There’s No One Trick”: Clarifying Critical Pedagogy in the Computer Science Classroom

Eric Mayhew

Submitted in partial fulfillment of the requirements for the degree of

Master of Science

School of Computer Science

McGill University

Montreal, Quebec, Canada

April 2023

Abstract

To counter the unintended harms of technology in society and unlock computing's potential to actualize a better world, Computer Science (CS) education needs to go beyond adding token ethics modules to CS curricula and rethink the pedagogical practices that lead to harmful technology. Critical pedagogy (CP) is a long-standing pedagogical tradition that aims to re-envision power structures in the classroom and foster a sense of agency and empowerment in students to change society for the better. Unfortunately, CP has been relatively underutilized in CS education. To expand CP in CS, this thesis goes beyond theoretical ideas of what CP is by providing an ontology of CP. To this end, we interviewed 13 computing educators who identified as being influenced by CP. In these interviews, we asked participants about their understanding of what constitutes CP and about the teaching practices they employ in their application of CP ideals. In the qualitative analysis, we identify four themes that our participants define as being fundamental components of CP: (1) meet students where they're at, (2) rethink student-teacher dynamic, (3) teach about power and how to identify power, and (4) take action and be reflexive. To provide a more in-depth description of CP, we showcase three vignettes from our interviews that give a rich description of CP in practice. Each vignette highlights a unique vision of CP centered around a respective goal: raising students' critical consciousness to see structures of oppression, helping students learn technology that supports their activism, and changing what it means to do computer science by integrating social and political forces. By providing concrete definitions and examples of CP, we hope to provide educators with inspiration for their own practice.

Résumé

Pour contrer les préjudices imprévus de la technologie au sein de la société et libérer le potentiel de l'informatique en vue de créer un monde meilleur, on doit faire plus que simplement ajouter pour la forme des modules sur l'éthique aux programmes d'enseignement en informatique et repenser les pratiques pédagogiques qui mènent à des technologies nuisibles. La pédagogie critique (PC) est une tradition pédagogique de longue date qui vise à repenser les structures du pouvoir dans les salles de classe et à encourager la capacité d'agir et l'autonomisation des étudiants afin de changer la société pour le mieux. Malheureusement, la PC a été relativement sous-utilisée dans l'enseignement de l'informatique. Afin d'étendre la PC à l'informatique, la présente thèse va au-delà des idées théoriques de ce que constitue la PC en présentant l'ontologie de la PC. À cette fin, nous avons interviewé 13 éducateurs en informatique qui ont dit être influencés par la PC. Durant ces entrevues, nous avons interrogé les participants sur leur compréhension de ce qu'est la PC et sur les pratiques d'enseignement qu'ils emploient pour appliquer les idéaux de la PC. Dans l'analyse qualitative, nous dégageons quatre thèmes que nos participants définissent comme étant des composantes fondamentales de la PC : (1) se mettre au niveau des étudiants ; (2) repenser la dynamique étudiants-enseignants ; (3) enseigner à propos du pouvoir et des façons de reconnaître le pouvoir, et (4) passer à l'action et faire de la réflexion. Pour fournir une description plus détaillée de la PC, nous présentons trois vignettes de nos entrevues qui proposent une description riche de la PC en pratique. Chaque vignette met en lumière une vision unique de la PC axée sur un objectif distinct : approfondir la conscience critique des étudiants pour qu'ils remarquent les structures de l'oppression ; aider les étudiants à apprendre la technologie qui soutient leur activisme ; et changer ce que faire de l'informatique signifie en intégrant les forces sociales et politiques. En donnant des définitions et des exemples concrets de PC, nous espérons inspirer les éducateurs dans leur propre pratique.

Acknowledgements

There are a number of people I'd like to thank in writing this research process. Although theses are presented as the culmination of an individual's work (*thanks liberalism*), they in fact are made with the support of many people and groups.

First and foremost, thank you to the participants who volunteered their time to talk to me about their practice. I thoroughly enjoyed our conversations. Having reviewed the interviews several times over, I can say with confidence you are all exemplary teachers who care deeply about your practice. Your teaching is fascinating and has given me so much to think (and write!) about. Reviewing all your different perspectives has been a deeply enriching experience for me.

Additionally, I thank my Social Studies of Computer Science labmates who assisted me during the coding of my many interviews. To Hana Darling-Wolf, Emma MacKay, Anna Ma, and Horațiu Halmaghi, you all consistently helped me code my interviews throughout the years. Thank you so much for the help and the friendships that formed along the way. You all made me feel like I was part of something bigger than myself.

I would like to pay particular attention to the help Anna Ma has helped me during my Master's and thesis writing. Anna, I never could have foreseen the touching personal and professional relationship that blossomed over the course of our studies. You have shown me many times over you are a patient, listening friend, and an intelligent, capable researcher. Thank you so much for the time and labour you put into helping me throughout my research process.

I would also like to say thank you to the friends who spent many hours co-working with me online or in person, listened to my half-baked thoughts, and proofread my work as I muddled through my research. To Kelly, Jess, Kris, and Anna, our co-working sessions helped keep me going and always brought a smile to my face. Additionally, I also thank Callan Ross-Sheppard and Jacob Errington for reviewing my thesis.

On to my supervisors: I sincerely thank Allison Gonsalves and Elizabeth Patitsas for their support in my research. Allison, you took me on as a student at a busy time for you and at the end of my degree. Thank you for extending your capabilities to include me in my final push when I needed it most.

And Elizabeth, I'm not entirely sure where to begin. I never could have guessed that taking that CS education class in Fall 2018 with you would go on to completely change the course of my life. At the time of taking that course, I didn't know what I wanted to do and felt dissatisfied with teaching. Working with you has raised my consciousness of the ways I could use teaching to make a positive change in the world. This has given me a purpose that guides me in this chaotic and disorderly world. It has been a privilege and honour working with you!

And finally, I thank Jacob Errington, my partner. Jake, you have been the foundation, the rock that has enabled me to go so much further than I could have imagined. You introduced me to CS and patiently taught me everything I know, starting with one of our earliest dates when you decided one of the first things I should code is a regex engine in Haskell. Needless to say, I learned a lot from you. And you continue to play a huge role in supporting me and my achievements, whether that's listening to my rants, insights, and epiphanies, or making me a warm meal and tea to enjoy after a long day of writing.

Contents

Acknowledgements i
List of Figures vi
List of Tables vii

1 Introduction **1**

1.1 Motivation 1
1.2 Literature on socially conscious CS education 3
1.3 Pedagogy 6
1.4 Theoretical Framing 7
1.4.1 Feminist Epistemology and The God Trick 7
1.4.2 Rendering Technical 9
1.4.3 Cultural Hegemony 10
1.4.4 Hacker culture and CS culture 11
1.4.5 Constructionism 12
1.4.6 Pedagogies of Resistance 14
1.4.7 Critical Pedagogy 14
1.5 Research Questions 18

2 Methods **20**

2.1 What is qualitative research? 20
2.2 Data collection 21
2.2.1 Participant recruitment 21

2.2.2 Participant background . . . 21
2.2.3 Interview format and questions . . . 21
2.3 Qualitative Analysis . . . 23
2.3.1 Coding . . . 23
2.3.2 Grouping codes . . . 23
2.3.3 Selecting examples for Chapter 3 . . . 24
2.3.4 Member checking . . . 26

3 Examples of CP in the CS classroom 27
3.1 P3: Raising students' critical consciousness . . . 28
3.2 P12: Helping students escape the punishment of poverty . . . 30
3.3 P5: Changing what it means to be a computer scientist . . . 31
3.4 Discussion . . . 34
3.4.1 Selecting vignettes . . . 34
3.4.2 Hegemony revealed from vignettes . . . 35
3.4.3 Limitations . . . 38

4 What is Critical Pedagogy? Creating an ontology from practitioner description 40
4.1 Plank 1: Engaging with students on their terms . . . 41
4.1.1 Seek to understand students and their communities . . . 42
4.1.2 Build on a constructivist pedagogy . . . 44
4.2 Plank 2: Rethinking the student-teacher dynamic . . . 45
4.2.1 Acknowledge and redistribute the power of the teacher . . . 45
4.2.2 Teacher as facilitator . . . 46
4.2.3 Empower and challenge students . . . 47
4.3 Plank 3: Discuss power in the classroom . . . 48
4.3.1 Give students the tools to identify power . . . 49
4.3.2 Discuss power and its relationship to CS . . . 50
4.3.3 Go meta: critically analyze the context of learning . . . 51

4.4 Plank 4: Take action . . . 52
4.4.1 Identify what students want and can do . . . 52
4.4.2 Take political action . . . 53
4.4.3 Be reflexive, cyclical, adaptive . . . 54
4.4.4 It's the combination that matters . . . 55
4.5 Discussion . . . 55
4.5.1 Hegemony in the four planks . . . 56
4.5.2 Limitations . . . 60

5 Discussion **61**
5.1 The many shades of CP . . . 61
5.2 Hegemonic forces in critical CS pedagogy . . . 62
5.2.1 Resisting rendering technical . . . 63
5.2.2 Tensions around constructionism . . . 64
5.2.3 Resisting or co-operating with liberalism? . . . 66
5.3 CP and related pedagogies . . . 67
5.3.1 CP and related social justice pedagogies . . . 67
5.4 Contributions . . . 68
5.5 Implications for CS educators . . . 69
5.6 Limitations and future work . . . 71

6 Conclusion **73**

List of Figures

1.1 **Survey of change in various majors' commitment to global citizenship before and after completing their degree.** Notice how CS leads to the largest negative change in this respect. 2

1.2 **Computing professionals' knowledge of environmental and social impacts of hardware manufacturing**. Note the majority of computing professionals in this study are "unfamiliar" with the consequences of computing. 3

2.1 **Organization of participant interview themes early on in the analysis process.** At this point, the analysis is in person and structured around "10 steps" of CP. 24

2.2 **Final organization of participant interview themes.** Now elements of CP are organized in four columns, each one facet. This picture also contains the grouping of teaching interventions participants use when doing CP. . . 25

4.1 **Diagram of our ontology of CP.** The four planks attach directly to CP, and each plank is surrounded by its sub-components. 42

List of Tables

2.1 Participant background information 22

Chapter 1

Introduction

1.1 Motivation[1]

As technology grows in presence and importance in society, so too do its political and social ramifications. This reality is underscored by the seemingly endless emergence of racist and disruptive technologies that exacerbate ongoing social inequities. For instance, take the community project initiated by David Dao to curate a list of "current scary usages of AI" [37][2]. This list contains dozens of terrible applications of AI, with ever-growing categories to sort them all, such as categories of discrimination, misinformation, and use in the military. This list, however, showcases the tip of the iceberg, as it focuses only on AI, ignoring for example the terrible and growing environmental impacts of computer manufacturing [19]. The near ubiquity of electronic technology reinforces the importance of teaching our students to consider the social and ethical ramifications of their work.

Alas, our students continue to disregard ethical, societal, and environmental impacts on the basis that they are "just an engineer" [25]. Indeed, surveys of computer science graduates suggest, if anything, that a degree in computer science and technology makes students *less* inclined to consider the political and social ramifications of their technology.

[1] Parts of this section borrow from Dr. Patitsas' and I's forthcoming paper *Critical Pedagogy in Practice in the Computing Classroom*

[2] I realize this citation style might be unfamiliar with those outside of CS. Numbered citations are the standard in CS (and I find it easier to read myself)

A survey of 5,500 students from over 120 universities and colleges in the United States finds that graduating CS majors report less of a commitment to social justice than students in other majors (see figure 1.1) [93].

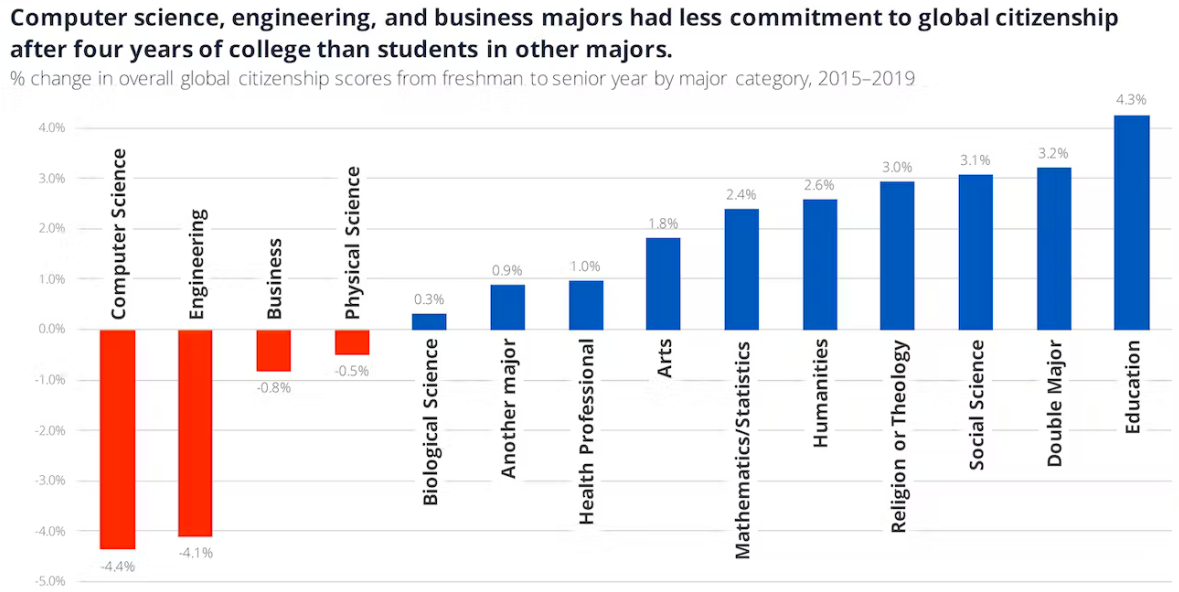

Figure 1.1: Survey of change in various majors' commitment to global citizenship before and after completing their degree. Notice how CS leads to the largest negative change in this respect.

Unfortunately, it's not just students that seem to ignore the social issues technology causes for society; programmers and computing professionals also appear ignorant of the social and environmental destruction brought about by computing technology manufacturing. Figure 1.2 illustrates the results from a survey of 135 computing professionals and academics, which found little to no awareness of the many harms of computing manufacturing, such as the widespread human rights abuses and environmental destruction resulting from computer manufacturing [19].

Clearly, the widespread disconnect between computing's consequences and its architects permeates the CS community more broadly. As technology continues to proliferate

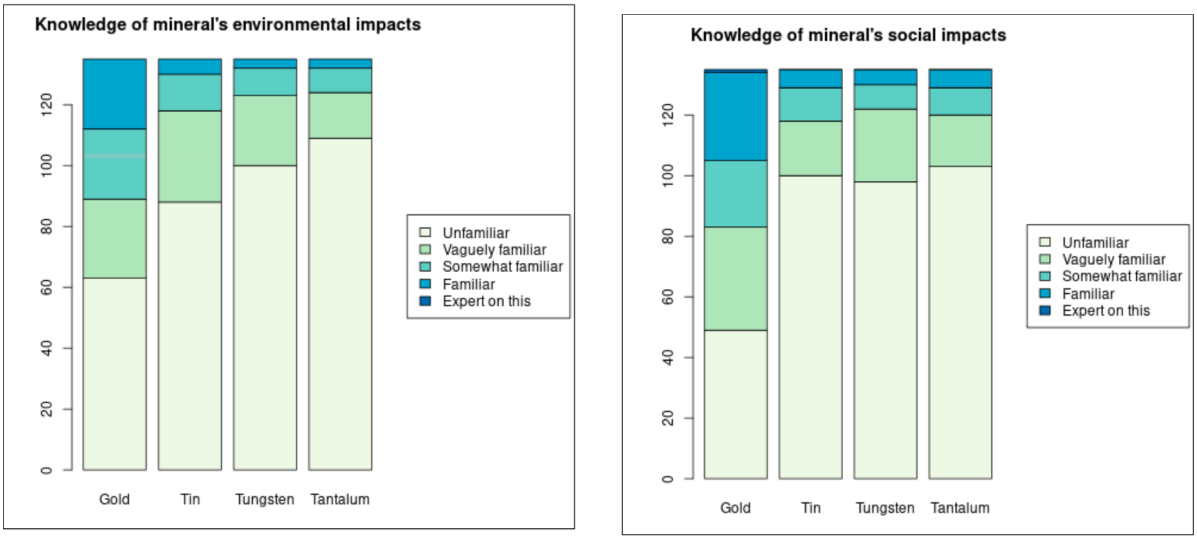

Figure 1.2: Computing professionals' knowledge of environmental and social impacts of hardware manufacturing. Note the majority of computing professionals in this study are "unfamiliar" with the consequences of computing.

in society, it is clear now more than ever that something must be done to improve the education programs that train the programmers that create and deploy these technologies.

1.2 Literature on socially conscious CS education

Computing educators have, in response, been working to bring social justice into the computing classroom. As early as 1972 [92], research communities in CS and engineering education begin concerning themselves with the societal implications of the way we teach CS [26,39,52,85]. Other researchers, like Tony Clear, stand out as being among (if not the) first to articulate the need and application of critical research methods in CS education research (CSER), as a way to address the lack of social justice in CS education [49].

Others, like John Impagliazzo, have advocated since the 1990s for the centering of history in CS curriculum, in an effort to "[view] computing systems in a broader social and historical context" [54, 61]. Historically, it is educators in the liberal arts college context

that were largely at the forefront of resisting an overly-technical CS curriculum. Since the 1970s, educators at liberal arts colleges have openly voiced fears of an overly prescriptive curriculum model of CS that focuses too largely on technical and scientific knowledge as this kind of curriculum is at odds with the goal of liberal arts colleges to provide students more general knowledge and skills [84]. Indeed, these efforts and criticism that CS curriculum is largely missing the social, political, and ethic implications of CS has been discussed for quite some time in the CSER community.

More contemporaneously, there has been a renewed interest in the introduction of more ethics courses and social considerations to CS programs. Consider for isntance, the growth of required ethics courses in large univiersities [117]. Within CSER, we see similar trends: social justice issues have proliferated in journals such as the ACM Transactions on Computing Education, which published a two-part issue on "Justice-Centered Computing Education" [1,2], in addition to the many calls for more research in this field [64,111]. Research in these circles has formulated a number of strategies to address the lack of social awareness and ethics. These range from adding ethics to class to CS programs, adding ethics to technical classes, or revamping CS programs altogether to weave social justice and ethics throughout.

What is perhaps the most common literature on ethical CS education revolves around research describing one-off ethics-in-CS courses. Consider Ferreira and Vardi's 2021 paper in which they share their experience of redesigning the program's ethical CS class to include "deep" ethical issues in tech, such as topics of social justice and technology's role in social inequality [47]. Another example comes from Purewal et al.'s paper *Embracing the social relevance: computing, ethics and the community*, which outlines the development of a CS ethics course with a final project built around sustainability and recycling [99]. This just skims the surface of many important CS ethics and social justice courses being offered at universities (for a review of these courses, see [48]). Outside of courses, there are many research articles describing these standalone CS ethics courses and how a teacher might develop one themselves [24,101,114].

Adjacent to the research on standalone CS courses, other research focuses on showcasing specific teaching practices one might employ in a CS ethics/social justice course. A number of articles share educators' experiences of developing new techniques to use in ethics or social justice CS classes, such as science fiction to make "ethical questions of technology development and use more vivid" [27]. Other research provides guidance on how to use mock trails [29], theatre [107], or gamification [24] to improve the teaching of these one-off ethics in CS courses.

Parallel to the development of these kinds of courses is the growing interest in embedding ethics and social justice into otherwise "technical" courses. For example, [55,76] assert that learning ethics alongside technical components can help instill in students that ethical and social considerations of technology must happen continuously and alongside the development of the technology itself. This is opposed to considering the social impacts of the technology separately and at the end of the development process — a step that is often forgotten. To this end, several articles in CS education venues share how educators go about integrating ethics or social justice into their technical classes. For instance, take Skirpan et al., who lay out in their work how they redesigned a "technical HCI course" to include discussion of ethics "without compromising core course material" [106]. In their paper, they discuss how they used workshops, guest speakers, and a final course project covering both HCI content and ethics in an integrated way. Efforts have also been made in intro programming classes [28,42,91,98], circuits and digital logic [72], and machine learning [102].

Some research also discusses changes to CS programs more widely to address issues of ethics and social justice in CS education. These program redesigns build off of the interest to weave social justice and ethics into CS technical programs more holistically. For example, Davis and Walker describe the process of readjusting a small, liberal arts college CS program to include issues of social justice throughout the program [40]. This includes changes to courses to include discussions of the social impacts of technology, as well as reading groups on social issues in computing and final projects that engage is-

sues of social justice. Grosz et al.'s paper, "Embedded EthiCS: Integrating Ethics Broadly Across Computer Science Education" is another example of research that shares a similar objective: redesigning a CS curriculum to include issues of ethics and social justice throughout the program [55]. Although the body of research on whole-program reorganization is smaller than that focusing on embedding ethics and social justice in a given course, this former research still remains an active and important line of research in the effort to incorporate meaningful social justice in CS programs writ large.

1.3 Pedagogy

The aforementioned research is highly important in the effort to introduce to students the unintended social consequences of technology, but it focuses almost solely on curriculum. The weakness in this focus is that curriculum construction is largely a policy-making decision, and the current culture of policy-making marginalizes the experience and expertise of educators. For this reason, effecting change through policy-making (i.e. changing curriculum) is a protracted and uphill battle for educators [86]. Crucially, this focus on curriculum omits important ways educators can advance social justice in their teaching without changing the content being taught.

Although curriculum decides *what* must be taught, educators have significant flexibility and power in *how* they teach. This flexibility includes, for example, the choice of a framing for the content (such as including a social and political framing), the distribution of power in the classroom, and the selection of values centered in the learning process. These elements of teaching and the autonomy of a teacher's practice outside of the mandated curriculum describe the *pedagogy* of the educator.

Recent developments in the sociology of education, largely headed by feminist scholars [87] highlight how pedagogy intersects more generally with processes of socialization that inform what is and is not of importance in a domain. It is pedagogy that "connects the apparently self-contained act of teaching with culture, structure, and mechanism of

social control" [6]. In particular, feminist pedagogies investigate "how particular relations are reflected and reproduced in schooling at a number of levels" [87], including racial, gendered, and ethnic identities. Indeed, scholars like Darder have warned how the control of knowledge works to cement asymmetrical power relations and to reinforce oppression [38]. This is what is referred to as the *hidden curriculum:* the lessons "which are learned but not openly intended" [77] to be learned in our classrooms.

If it is the case that pedagogy connects teaching with culture and structures, then understanding how these mechanisms work may avail to sociologists of education the levers that produce apolitical, socially unaware programmers. With this knowledge, educators and sociologists of education could thereby furthering our collective understanding of how to undo this production.

1.4 Theoretical Framing

This section outlines the framework that explains the lack of social implications in CS education and how education can correct this. To explore the lack of social consciousness in CS students, we rely heavily on anthropologists Samantha Breslin's thesis *The making of Computer Scientists* [22], which analyzes how CS education builds a particular social identity that omits the social and political consequences of CS to students. Large parts of this section reuse the theoretical framework used in a separate paper Dr. Patitsas and I wrote, *Materiality Matters in Computing Education: A Duoethnography of Two Digital Logic Educators* [80], which details our process of becoming aware of how our teaching replicates the apolitical framing of CS education. Additionally, this section will outline the theory that underlies the potential for education to counter these trends.

1.4.1 Feminist Epistemology and The God Trick

Building on ideas from Donna Haraway, Breslin shows how students are taught by authority that CS is objective, in contrast to many contemporary understandings of episte-

mology which assert that all knowledge produced by humans is situated in the context in which it was produced and limited by the perspective of the knower [58]. Indeed, for feminist philosophies of science like Haraway's, embracing the partiality of scientific knowledge is necessary for science to live up to its ideals of transparency and reproducibility [58,109]. Denying the partiality of science leads to distrust of science, especially by those who are actively harmed by its false claims of objectivity [96].

Haraway coined the notion of the "God Trick" to refer to the myth that the scientific process yields so-called "objective truths" about the world, creating knowledge that "sees everything from nowhere". However, since all knowledge is situated in the context of its origin, this "objectivity" promotes views from a particular standpoint (generally Western, male, abled, white, etc) as purportedly "universal truths".

Indeed, feminist approaches to scientific methodology stress that the importance of situating oneself in one's research, and being continuously reflexive about how one's standpoint affects the knowledge one produces [59]. In line with this methodology, the following section outlines a standpoint statement from myself, as well as my reflections throughout the thesis.

Who am I?

I'll start by saying I find this actually quite difficult to write this. As a junior scholar, I'm just getting the hang of the academic voice expected of me when writing research — now going back to being more personal is rather difficult.

I am a gay white man, currently finishing my Master's in CS, with Elizabeth Patitsas and Allison Gonsalves as my supervisors. Before my Master's, I graduated with my Bachelor of Education in 2019 from McGill. During my bachelor's and as a side hustle, I taught as much as I could as teaching gives me purpose. At the tail end of my B. Ed, I met my wonderful boyfriend Jake who showed me the powerful and exciting world of computers. I was always suspicious of injustice and inequality in society, and Elizabeth helped me see how these problems crop up in computing too.

When I first started this research journey, I remember reading about social justice CS education. I found this research lacking in helping to conceptualize how I should change my practice to be more justice oriented. I found many papers describing at a theoretical level what critical teaching "should" look like but I found examples of what *not* to do in these articles more representative of my teaching at the time. This became the motivation for my research: setting out to articulate a radical, anti-oppressive way of teaching. Being a young professional at the start of my teaching career, this research comes at a good time for me. Now I can reflect and consider how to make my teaching better from the outset before I get too comfortable.

1.4.2 Rendering Technical

Finally, Breslin uses Tania Murray Li's work on hegemonic processes in global development [70]. Li defines "rendering natural" as a hegemonic process in which a dominant group's practices are made to be seen as "natural" by other groups — such as how exploitation, capitalism, environmental destruction, extractivism, and neocolonialism are presented in the global development community as natural and unquestionable parts of human society and progress.

Li also defines "rendering technical" as the hegemonic process by which development and environmental issues are stripped of their context and reduced to technical problems in need of technical solutions. Technical solutionism acts as a hegemonic norm in many environmental spaces where local context and political action are neglected in favour of shiny technological solutions [15,73].

Breslin documents how a CS educator teaches students to render the world technical: to strip away all context of "real world" problems and see the world *only* in terms of abstract computational problems, ready for computational solutions. This stands in contrast with context-appreciative design philosophies such as Soft Systems Methodology, in which the modeler fully immerses themself in the local context, abstracts it to

form a model, and finally de-abstracts the model to meaningfully integrate it into its local situation [32].

Rendering technical serves to socially construct CS as something that is separate from the "real world" that it interacts with [22, 57]. The separation of CS from society and the environment is thus a hegemonic norm in the discipline [7, 13, 16, 34, 74]. Rendering technical also serves to promote capitalist notions of production [14, 71, 89]. Together, these result in the "I'm just an engineer" phenomenon [48, 95] that in turn leads to computer scientists viewing sustainability and social justice as beyond their scope. Indeed, Mann et al. find that by surveying CS students about their moral development that 79% percent of CS students agreed with at least one of these statements: "Computing is largely theoretical or technical - with little consequence", "There is no room in business for soft things like ethics, if your competitor does it then you can", and "Business is a special case; the ethics are different to personal life" [75].

1.4.3 Cultural Hegemony

Breslin also draws on Antonio Gramsci's notion of *cultural hegemony* to demonstrate that the enculturation of CS students leads them to buy into the dominant group's ideology. To Gramsci, cultural hegemony "is a social condition in which all aspects of social reality are dominated by or supportive of a single class" [41]; through culture, the dominating class can manipulate other groups to buy into the dominant group's beliefs and interests [51]. For example, "in a capitalist society, journalists, the media in general, and professors create the sense that capitalism is efficient, egalitarian, natural, and so on [...]", making it difficult or laughable to envision a world without it [51].

In the CS context, Breslin describes how students' adoption of an apolitical CS identity is hegemonic, which includes rendering technical not only the world but also the student's self-perception. In her analysis of hacker groups, Breslin notes how students often compare themselves to one another, particularly along the lines of passion, which

manifests as a desire to self-learn CS outside class. This dovetails with prioritizing values of independence and independent learning in this CS identity.

1.4.4 Hacker culture and CS culture

The hegemonic personhood Breslin articulates connects to the liberal hacker ideology that characterizes the culture of CS more broadly. In their work *Hacker Practice*, Coleman and Golub explain how hacker culture and liberal ideology overlap and interact. Both the cultural ideology of liberalism and hacker ethic are committed to a number of shared values and outcomes, such as protecting property and civil liberties (e.g. free speech), individual freedom, equal opportunity, and meritocracy. To Coleman and Golub, "liberalism works as one important context by which hackers make sense of their selves and their world as well as justify the tools they produce" [36].

Commitments to a liberal ideology like free speech and individualism run deep in the CS context and are perhaps best exemplified in the Free and Open Source Software (FOSS) movement, a unique characteristic of the CS context. What is meant by "free" here is explained by Richard Stallman, a central figure in the FOSS movement: "Free software is a matter of liberty, not price. To understand the concept, you should think of free as in free speech, not as in free beer" (cited in [36]). "Open-source" refers to the fact the source code of any FOSS software is free to study, distribute, and modify — that is, free from constraining copyright laws.

Individualism and meritocracy similarly play an important part in adopting a CS identity in the FOSS community. Like Breslin describes, CS hegemonic personhood centers on passion and a do-it-yourself attitude. This is further corroborated and expanded on in Dawn Nafus' study of FOSS culture [88]. In her study, she describes how programmers cultivate a CS identity based on individualism and meritocracy: "Many of our research participants began reading computer manuals and books from a very young age. Yet they recognized this only as an independent auto-generation of knowledge between person and machine, not a communication between author and reader". This goes to show

how deeply connected narratives around individual meritocracy are to the adoption of CS identity, which lends itself to Breslin's definition of hegemonic CS identity.

1.4.5 Constructionism

From the hacker culture emerges a pedagogy common in CS called *constructionism*. Constructionism is a philosophy of learning that advocates for the unfettered student-centered discovery of content material through project-based learning. This pedagogy is credited to Seymour Papert, who coined this term in his best-selling book *Mindstorms* [94], originally published in 1980.

As Morgan Ames details in her exploration of hacker culture and constructionism, Papert was heavily inspired by MIT's "nascent 'hacker' culture" in 1964 [7]. This hacker culture is comprised of several tenets that left an indelible mark on Papert and became the cornerstones of constructionism. These tenets include providing students "freedom" to use computers as objects to play with, as well as judging hackers "by their hacking, not bogus criteria" like degrees, age, race, or position. It's clear how these tenets are based on the liberal ideology of freedom and meritocracy, making the connection between hacker culture, constructionism, and liberalism unavoidable.

Constructionism, therefore, focuses on *doing* to learn, instead of relying on traditional ways of learning often used in institutional schooling, such as lecturing. Papert's aims are to create software that can be used as a medium of expression for young learners, thereby enabling young children's play, and thus, learning. To this end, Papert created the first-ever programming language designed to help students learn to code, called LOGO. Being the first programming language to center learning (as opposed to programming languages for general use) LOGO represents a highly important early contribution to CS education.

Indeed, this teaching philosophy remains dominant in CS education, despite having lost popularity in adjacent domains [7]. Many technologists resonate with constructionism's claims that school-based learning is stifling and students are best left to learn on

their own [7], as this is how many programmers learn CS. Educators, eager to share the joy of self-teaching CS, hope to bring this experience to their students using constructionist teaching approaches. By providing learners with unfettered access to computers, students are free to learn by themselves and learn the skills to hack and code that will go on to be the metric by which they are judged. And from this, we see how the hegemonic CS identity articulated by Breslin is reproduced through constructionism.

However, this way of learning may not represent how the average learner learns best; this is a major weakness of this pedagogy. Indeed, Papert's work makes the assumption that a technically "precocious young white boy" is representative of learners writ large [7], without regard for the gendered, racialized, and classed dynamics involved in learning CS. Constructionism fails to take into account identity formation in CS, in particular how non-white men students often do not see themselves as technologists.

By focusing on self-learning and passion while disregarding the social and historical elements of participation in CS education, a connection between constructionism and the hegemonic CS identity Breslin describes is clear. The conditions of constructionism, with a focus on out-of-school learning and self-directed learning, demand the adoption of a hegemonic CS identity that shares many values. However, constructionism's disregard for the social factors that lead to the exclusion of minorities from CS neutralizes this pedagogy's potential for ameliorating the inequalities in technology more generally.

Constructivism

Construct*ionism* is not to be confused with the similarly named learning theory, construct*ivisim*. Constructivism is a seminal learning theory in education research posited by Piaget in the 1920s. This learning theory asserts that knowledge is not "transmitted" from teacher to learner, instead learners *construct* knowledge [4]. This construction of knowledge is an active process, not a passive one, as a learner builds on prior knowledge and experiences to incorporate new information. Constructivism has had huge impacts

on the field of education, inspiring the creation of many other pedagogies and learning theories, such as constructionism – hence the similarity in naming.

1.4.6 Pedagogies of Resistance

Fortunately, no hegemony is ever complete [31], and pedagogy can offer a form of resistance to hegemony. As la paperson writes in their book *A Third University is Possible*, the education structures, such as those that manufacture hegemonic identities, can be reconfigured in new assemblages that offer an alternative to this production [118]. Education then can be an essential tool to undo apolitical CS education and center the social.

Indeed, there exist pedagogies that are aimed exactly at re-conceptualizing teaching and learning and that are more just and socially aware. As mentioned in section 1.3, pedagogy is an important tool for shaping student identity through processes of socialization. Therefore, pedagogy can also equip students with the tools to enact a more socially just society or create space to center students from non-dominant backgrounds. Some pedagogies of note with this goal in CS education are culturally relevant pedagogy (CRP), ethnocomputing, and finally, critical pedagogy (CP), all of which will be discussed here.

1.4.7 Critical Pedagogy

Critical pedagogy (CP) is another pedagogy worth consideration in a CS education focused on social justice. Although it remains underutilized, CP shows promise in the effort to center issues of social justice and inequity in STEM education. This pedagogy originates in Paulo Freire's foundational work *Pedagogy of the Oppressed* [50], which was developed for adult literacy education in 1960s Brazil, at a time when a passing a literacy test was necessary for citizens to vote.

Freire criticizes what he termed the "banking model of education", in which educators aim to deposit "coins of knowledge" into their students' "bank accounts" (brains). In the banking model, students are passive recipients, receiving discrete tokens of "knowledge".

Societal power structures are reinforced through the hierarchy of the all-knowing teacher and the student-as-receptacle.

In contrast to the banking model, Freire envisions a structure in which students and teachers collaborate to learn together, where educators aim to impart a critical consciousness, an ability to identify power structures and reimagine them. Together, students and teachers expose and disrupt oppressive power structures in society, e.g. patriarchy, white supremacy, and ableism. It is common practice in CP for educators to raise student consciousness of the often hidden ways by which power operates in students' day-to-day lives and supports students in their efforts to resist these structures.

Critical pedagogy has spread into a variety of contexts and domains [23]. While there has been ample work in applying CP principles in other domains such as math education [12,68,108], medical education [79], science education [103] and music education [3]. CP in fields related to CS, like science [104], math [12], and physics [35] have been critical of these domains are portrayed as "hard sciences" with undisputed claims "about the objectivity and universality" [12]. Unfortunately, CP remains relatively uncommon in CS [34, 63]. Yet at the same time, digital literacies are becoming increasingly important as computing technologies play increasingly relevant roles in government operation [46], information distribution in society [17], and the gathering and commercialization of user data [25].

Criticism of CP

Despite the emancipatory and justice-oriented aims of critical pedagogy, critical pedagogy is criticized for overemphasizing doing "good" without seriously considering how CP perpetuates inequality and injustice. In the collection of essays *Rethinking Freire*, Bowers highlights how CP emphasizes "rescuing" the illiterate other without realizing how this emphasis of literacy implicitly prioritizes Eurocentric systems of knowledge [21]. Generally, pedagogies that focus on "doing good" are vulnerable to being hollowed out, where well-intentioned teachers trying to make a difference adopt what sounds like an

emancipatory pedagogy that in fact might not be effective in bringing about change. For instance, awareness raising, a key tenet of CP, can become an overemphasized objective. Simply put, raising awareness is insufficient on its own to make meaningful social change. Instead, awareness raising must be complemented with clear calls to action if social change is the goal [33].

Another critique of the CP literature has been its emphasis on theoretical ideals and considerations of power structures; rather than concrete, tangible things that educators can use in their own teaching. Many researchers, such as Carlson [30], Gur-Ze'ev [56], Van Heertum [112], and perhaps most notably, Apple [9–11] have criticized CP and related domains of critical theory for "[sharing] too firm a commitment to the first half of their [name], forgetting that critique alone has never led any revolution" [105]. Take Tower's keynote address *Reinventing the university: Visions and hallucinations*, which "outlines a range of literature [on CP ... which] offer little in terms of concrete implementation strategies" [81]. There is a small and growing literature on CP in CS, and indeed much of it has been theoretical, such as [34, 64, 111].

Culturally relevant pedagogy (CRP)

CRP seeks to incorporate the cultural backgrounds and experiences of students into the learning process. This approach recognizes that students come from a wide range of cultural backgrounds, and aims to use this culture as a vehicle for learning. Ladson-Billings articulates three criteria for CRP: "(a) Students must experience academic success; (b) students must develop and/or maintain cultural competence; and (c) students must develop a critical consciousness through which they challenge the status quo of the current social order." The first two elements work in tandem: by understanding a student's cultural identity, their culture can be centered in the learning process, thereby promoting academic success and maintaining the students' connection to their cultural background. The last element includes giving students the knowledge to identify inequalities in society and critique them. As Ladson-Billings so eloquently argues: "If school is about preparing

students for active citizenship, what better citizenship tool than the ability to critically analyze the society?" [65].

In the CS context, Yuen et al. have articulated what CRP would look like in this domain [119]. In their adaptation of CRP for CS, they arrive at five requirements:

1. "Opportunities for students to be successful in CS;
2. Learning activities that serve multiple contexts by integrating students' home lives into the classroom and vice versa;
3. Authentic learning activities that integrate computer science across the curricula to show the interdisciplinarity of computer science;
4. Empowering students to understand their sociopolitical positions in the computer science field, and how to transcend any forms of oppression and challenges that currently exist for minorities in computer science; and
5. Mentoring opportunities for students of underrepresented minorities, especially from mentors who are successful computer scientists who share similar cultural and linguistic backgrounds as the students."

Although this is an important theoretical discussion of CRP in CS, it remains just that: theoretical. CRP shows much promise for the CS context, yet more research in the application of this pedagogy is limited. In particular, of interest would be to see how current pedagogues are employing CRP. This would ideally flesh out such ambitious goals as enabling students to "transcend any forms of oppression [...] that currently exist for minorities" in CS.

Liberal arts and ethno-computing

In CSER and CS education, two unique strands exist to address the lack of socail awareness in CS eduation. Starting with liberal arts, many researchers in the community have

historically called for a more socially aware CS curriculum, in line with the general learning goals of liberal arts colleges [26, 52]. Consider Goldweber et al., who articulate a "Computing for Social Good" curriculum in their research agenda, which centers on issues like climate change in their CS education [53]. Liberal arts colleges, with their goal of providing students well-rounded knowledge bases to students, have historically called for a more social-sentistive CS curriculum. Another important and unique pedagogy in CS education research called ethnocomputing, which has its origins in ethnomathematics. This pedagogy situates in the cultural context of the learner, thereby helping students explore the relationship between computational ideas in their cultures and ideas [44]. This connects the CS knowledge with the knowledge from students' cultural and ethnic backgrounds as a way to support their conceptualization of CS knowledge, not mearly as a way to draw students with marginalized identities into CS. This itself as a unique way CS pedagogies seek to address issues of inequality and underrepresentation in CS education. What makes these pedagogies different from CP however is CP focuses largely on political activism with the explicit goal to empower students to create change in society.

1.5 Research Questions

We hold the potential for CP to address the blind spots in the dominant framing of CS knowledge. We believe, for instance, the lack of clarity on what CP looks like concretely can be addressed. Indeed, this is what motivates the research for me, as when I first learned about CP I was left confused about how I could implement this pedagogy in my own teaching. As a result, I (with the help of my supervisors) set out to collect and organize examples of CP being done in the CS classroom to inform my own practice.

Therefore this thesis looks to provide a descriptive approach to how CP is being employed in the classroom. Our objective is to contribute to the literature a set of illustrative, tangible practices, a CP educator can do. From this, we also contribute an ontology of CP in CS. By ontology, we mean a categorization of the properties and concepts our partici-

pants described CP as, and how these categories relate to one another [115]. To do this, we formulate two research questions:

- For CS educators who are inspired by CP, what instructional interventions do they describe using when teaching CS critically?
- For CS educators who are inspired by CP, what elements, concepts, and considerations do they describe as constituting CP?

We interviewed 13 CS educators identifying as being (at least) influenced by CP. We asked participants questions about concrete teaching practices and examples of their work. We share select excerpts from our interview data that we think richly highlight *how* educators go about teaching CS critically. Our goal is to show educators wishing to incorporate criticality into their teaching some examples so they may inform their own practice.

Chapter 2

Methods

The empirical basis of this thesis, presented in Chapter 3, is entirely qualitative. Given quantitative methods are more common in computer science, this chapter aims to familiarize readers with qualitative research and the methods used in this study. This section reuses large portions of text from the forthcoming paper I wrote with Dr. Patitsas, *Critical Pedagogy in Practice in the Computing Classroom*.

2.1 What is qualitative research?

Qualitative research aims to "achieve an understanding of how people make sense of their lives, delineate the process (rather than the outcome or product) of meaning-making, and describe how people interpret what they experience" [83]. Given that this study aims to understand how critical pedagogues understand their practice and experience teaching critically, qualitative research methods provide a closer fit to address our research questions. Unlike quantitative research methods, which are more common in computer science research, qualitative research embraces subjectivity in data.

With no known existing qualitative (or quantitative) research on critical computer science educators, our goal in this study is to begin this line of investigation. We chose to

speak with educators influenced by critical pedagogy with teaching experience to understand this population.

2.2 Data collection

2.2.1 Participant recruitment

We began recruiting participants in 2019, using social media (Twitter), cold emails, and mailing lists related to computer science education. From there, more participants were included in the study through snowballing, i.e. by asking participants in our study if they would recommend anyone to be interviewed next. The only requirement for participation, as described in the call for participants, is experience teaching computer science and self-identify as a being (at least) influenced by critical pedagogy.

2.2.2 Participant background

Participants for this research came from a variety of backgrounds. Our participants included 8 professors, 2 doctoral or postdoctoral researchers, 1 high school teacher, and 2 educators in informal contexts. All but one of our participants are white and all but two participants are from the United States, Canada, or Europe. Of our participants, 10 identified as men, 2 as women, and 1 as non-binary. The following table lays out the background of our participants. Participants are given an anonymous identifying number (e.g. 1 through 13), in order of who is interviewed first.

2.2.3 Interview format and questions

I, Eric, conducted all 13 interviews with participants. The interviews were semi-structured and each interview was conducted via the video-conferencing software of the participant's choice, with each interview lasting roughly one hour. Interviews were conducted between July 2018 and July 2020. Initial transcriptions were generated from the interview

ID	Position	Setting
P1	Not for profit director	Informal, primary and secondary
P2	Entrepreneur	Informal, primary and secondary
P3	Former high school teacher	Secondary
P4	Professor	Public research university
P5	Professor	Private research university
P6	Professor	Public research university
P7	Professor	Public research university
P8	Professor	Public research university
P9	Professor	Public research university
P10	PhD Student	Public research university
P11	Post-doctoral researcher	Public research university
P12	Professor	Private research university
P13	Professor	Public research university

Table 2.1: Participant background information

recordings using transcribing software, followed by manual correction of the transcripts by the first author.

The questions in the interview were intended to understand how participants define critical pedagogy as well as to extract illustrative examples of how participants concretely teach CS critically. All participants were asked (at a minimum) the following questions:

1. What is your teaching philosophy?
2. What is your understanding of CP?
   (a) What is not part of your understanding of CP?
   (b) How did you first learn about CP?
   (c) Does your teaching philosophy connect to CP? If yes, how?
3. To you, what are the necessary elements of CP?
4. Let's go through each element of CP you listed one at a time. Can you elaborate on each item and explain to me how you incorporate this item into your teaching?

Often times, new questions were generated by the responses to those above. However, all participants answered the above questions.

2.3 Qualitative Analysis

2.3.1 Coding

Transcripts were analyzed using qualitative content analysis in order to find commonalities and differences across interviews. Each transcript was manually coded with consensus by myself and with help from the members of the Social Studies of Computing Lab. Over the years of analysis, multiple coders participated in the coding. Each coding session had an average of three coders. All coders who participated either had a strong background in education, computer science, or both. In the interest of time, the final four interviews coded were done independently by myself.

During group coding sessions, interviews were processed one question at a time. To facilitate coding, each question and answer were broken down into smaller excerpts. Excerpts were coded sequentially. During the coding sessions, each member read the same excerpt from the transcript. Then, members discussed what codes they identified in each excerpt. These emergent codes were refined until we reached a consensus.

Initially, we wrote the code on a sticky note and posted it on a blank wall. However, halfway through our analysis, our work transitioned online. Old sticky notes were manually migrated to a virtual whiteboard and all future coding proceeded online.

2.3.2 Grouping codes

After each interview was completely coded, all codes were grouped using affinity mapping. Only after the first three interviews were coded in December 2019 did clear themes emerge. As new interviews were coded and added to the mapping, themes were reworked. Oftentimes, the process of integrating new codes into the established structure resulted in the reorganization of themes, making the analysis an iterative process. For example, Figure 2.1 shows the main macro-level tenets of CP that emerged from our early data analysis (what we called at that time the "10 steps of CP"). Subsequent iterations

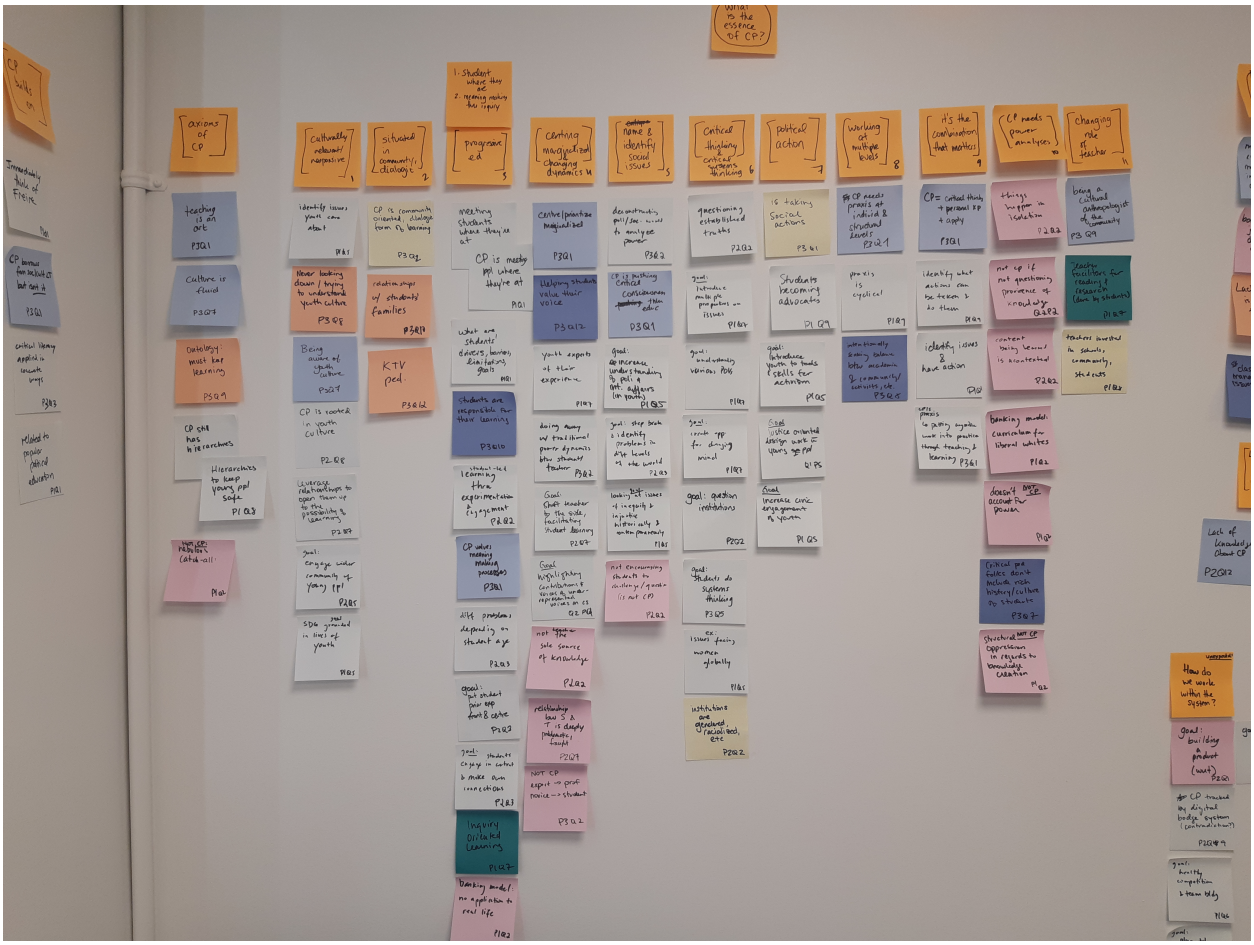

Figure 2.1: Organization of participant interview themes early on in the analysis process. At this point, the analysis is in person and structured around "10 steps" of CP.

of our data (pictured in figure 2.2) resulted in a reconfiguration of the same high-level themes, into four planks of CP.

2.3.3 Selecting examples for Chapter 3

For the purpose of the results presented in Chapter 3, we select three participant interviews to highlight contrasting visions of critical computer science education. It was an iterative process to decide which participants to select to showcase divergent visions of critical CS pedagogy. Initially, the first author reviewed all interview transcripts and selected six short and illustrative excerpts. Then, members of the authors' lab reviewed the excerpts and decided which three would be most useful for educators to learn from.

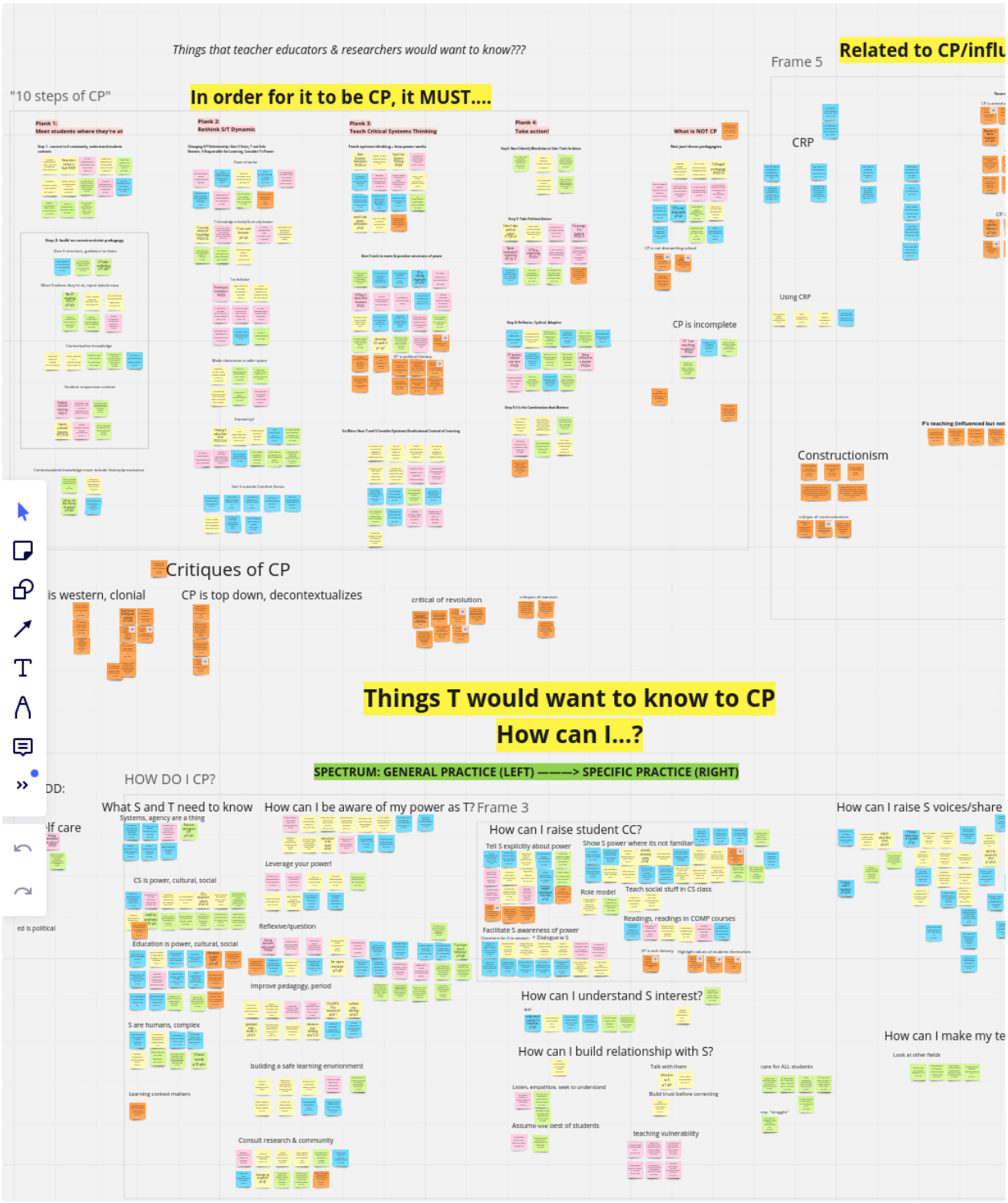

Figure 2.2: Final organization of participant interview themes. Now elements of CP are organized in four columns, each one facet. This picture also contains the grouping of teaching interventions participants use when doing CP.

We chose these three excerpts for several reasons. First, they come from different educational settings, e.g. different levels or student demographics. Second, these excerpts demonstrate a variance in the educator's goals, such as raising consciousness or making a tangible change in the students' community. Finally, these excerpts are very *vivid:* they are succinct and radical examples of critical CS education.

2.3.4 Member checking

To ensure the validity of our work, we have member checked each vignette with its respective participant. Member checking is a technique in qualitative research where findings (in this case, the vignettes and our interpretations thereof) are shared with the respective participant for feedback. Their feedback serves as a check on the viability, fairness, and accuracy of our interpretations.

Chapter 3

Examples of CP in the CS classroom

Our analysis suggests multiple visions and approaches to critical pedagogy in computer science. From our data, several teaching philosophies and practices emerge under the name of critical CS pedagogy. To organize our findings, we present two chapters on our results. This chapter will provide a fine-grain presentation of our interview findings, based largely on the forthcoming research Dr. Patitsas and I published on these results titled *Critical Pedagogy in Practice in the Computing Classroom*.

To illustrate CP in a visceral way, this section will draw on three vignettes of critical CS pedagogy. We chose these three vignettes because they offer to contrast and striking ways of doing CP in the CS classroom that show a variety of approaches to teaching CS this way. The goal is to provide a highly detailed selection of the currently employed approaches in CP. By providing practical vignettes of critical CS pedagogy, we hope that other educators interested in introducing criticality in their practice can choose what is appropriate to their context.

We identify a *goal* for each of the three approaches to critical CS pedagogy highlighted by these vignettes: raising student awareness to systems of power (P3), helping students regain access to unfairly seized resources (P12), and changing the disciplinary norms of computer science to be more critical (P5). All but one of these goals are among the stated

teaching goals of the respective participant; each goal should not be understood as the sole goal of each participant's teaching.

3.1 P3: Raising students' critical consciousness

P3's vignette comes from their experience teaching Exploring Computer Science, a K-12 curriculum from the U.S. P3 taught in a high school context, taking over a course traditionally taught using Scratch, and saw an opportunity to introduce criticality into this course:

> I felt that from critical training, many of the projects when I started working were getting students to recreate arcade-style games or shoot 'em up games, and I felt like we could use this similar approach but add in this criticality into this unit.

To impart criticality to their teaching, P3 chooses to emphasize raising students' awareness of the power structures they belong to and their positions therein. To do so, P3 focuses on the "history of [the students'] community":

> What I felt the best way to do [criticality in this course] was to first delve into a better understanding of the history of their community - the hyper localized community where they lived.

By narrowing in on the local history, P3 enables students to understand how systems of dispossession affect the students' communities and lives. P3 "created this whole curriculum where we analyzed the social, historical, and political processes that shaped the way their community is now." Concretely, the teaching practices P3 employs in this curriculum include:

- critically analyzing popular media such as music, kids videos, music videos, and advertisements;

- watching documentaries;
- having students conduct interviews with community members to learn their community's history; and
- facilitating discussions between students around stereotypes and dominant narratives about the student's identity.

P3 reports this approach to teaching CS confused some students. According to P3, some students asked after the first few weeks of classes, "How come we're still not on the computer? Why are we learning all this stuff?" For P3, raising students' consciousness is a requirement to add criticality to the course.

The course's final project is to create an interactive fictional story (a.k.a. choose-your-own-adventure story) that provides a counter-narrative of the students or their community, to be shared on the Scratch online platform. By publishing their stories online, the students resist the dominant narratives about their identity and provide some alternatives.

Given the students' community is historically dispossessed, P3 knows that many students already "had to contend with multiple layers of oppression." P3 describes how this project gives students an opportunity to reflect on their experience belonging to oppressive power structures. Students' stories discuss:

> what is it like to be an undocumented immigrant, what's their life like - navigating daily realities, some other students discussed the realities of growing up in a high-crime community and what the day-to-day decision-making processes that one has to make.

This sharing and reflection are happening at the same time as students have fun doing it according to P3:

> [students were] having a lot of fun because they're drawing, animating characters, coming up with stories, all while still learning coding. They're learning

> many aspects like condition statements, parallel execution, and systems thinking. They're bridging all these things together.

This approach to critical pedagogy, therefore, lies heavily in raising students' awareness of oppressive systems in their lives. The goal is to awaken students to the systems and power structures they belong to so they may resist these structures and consciously carve out a space for themselves in these structures.

3.2 P12: Helping students escape the punishment of poverty

P12's vignette details their experience designing a three-week, after-school robotics workshop for elementary students in Brazil. At the time of the workshop, Brazil was amid an energy crisis, with electricity in high demand. The increased cost of energy leaves many unable to pay their electricity bills, resulting in the revocation of their electricity access. P12, seeking to teach something both relevant to students' lives and helpful to them, designs the workshop to center on energy-saving devices students can use in their homes to conserve energy:

> When I got there, I said, "Okay, we're gonna do energy-saving devices, so you're gonna save energy in your bill, and also help the planet."

However, P12 quickly discovers, in the first lesson, that many of the students already have their access to electricity taken away:

> [After introducing the energy-saving devices] the kids were looking at each other [like] "what is this guy talking about?", and someone told me, "Oh, actually, we don't have energy meters in our house."

To P12, this is a "moment of shock and despair for me, because I thought I was going to be super relevant to them."

Determined to provide students with a meaningful learning experience, P12 discusses with the students what they would like to learn about. Students are quick to inform their

instructor that they would like to create resources for safely splicing electrical wires from streetlamps for domestic use. For many, illegally splicing wires from streetlamps serves as the only way to get any power in their homes to run kitchen appliances. However, this process is extremely dangerous and oftentimes fatal. What students are interested in doing is creating a documentary and leaflet for the community about how to safely create those illegal connections.

P12 reflects on this teaching experience as a "pivotal moment" in their development.

> it's just an example of, you know, sometimes you have your whole design done, and you spent tons of time preparing it and, but you don't really know, the people you're dealing with, right? So I think you always have to be, you know, open to that.

P12's teaching centers on giving students ample time to use the schools' technological equipment and to create projects they are personally interested in. This teaching style aligns closely with constructionism, a teaching philosophy P12 identifies with. Students are provided with guidance when they need it or counseling when they are unsure what to do with their free time.

What culminates from this workshop are tangible, hands-on resources for students and their community to resist the cruel punishment of poverty. This approach to critical pedagogy highlights a practical impact on students and their community, with a focus on liberating students from the unjust distribution of resources. Students and their communities are enabled to create and use tools to resist the oppression of poverty through this kind of critical pedagogy.

3.3 P5: Changing what it means to be a computer scientist

P5 is a university professor and shares their experience co-teaching a large CS1 course and facilitating one of the many recitation and lab sections. This CS1 course is unique

in that as students learn the technical skills for programming, they are also exposed to sociological or cultural studies readings.

> Early on in the class when students are learning what programming is and what CS we will be reading things like Judy Wajcman's *Feminism Confronts Technology*, or we'll use Tara McPherson's *U.S Operating Systems at Mid-Century* which are these really nice historical and critical takes on how did CS come to be what it is? Why do CS industry and classroom spaces look the way they do?

Lab time is divided between learning technical concepts and discussing the readings. Interestingly, P5 finds that freshman CS students do not struggle more than freshman sociology students who read the same material: CS students are just as capable of reading sociology literature. To facilitate student learning, P5 creates space in the classroom for students to wrestle with the challenging findings in the sociological literature. Indeed, P5 reminds students that "it's OK" to feel the readings are difficult, "just as coding is difficult". P5 motivates students to persevere through the reading by reminding them that this is an "important part of forming who you are as a student and individual."

One teaching technique P5 uses to continually expose students to a sociological perspective of computing is through assignments. One assignment, which aims to teach students about the use of dictionaries in Python, also uses a data set on Irish immigration into New York City from the 19th century to expose the shortcomings of computational thinking when applied to a social context. This dataset consists of the information that immigration admittors collected on each Irish immigrant. While learning to manipulate dictionaries of immigration records, students are also made to scrutinize how these immigration admittors' biases become encoded in the dataset. The goal of the assignment is to show students first that biases can easily become reflected in data and second that the pattern recognition abilities of computers can be used to help unearth these biases.

To scaffold students in understanding bias in datasets, a part of this assignment involves students sorting the data set by each immigration admittor and asking students

to compare how different admittors had different standards by which immigrants are judged, such as whether a given immigrant was "diseased" or not. To P5, this serves two purposes. It shows students both how data collectors' biases become encoded and naturalized in data sets and how students can use computational methods to identify inequalities:

> And so we talk to them both about problems with turning social issues into computation models but also if you use it kind of carefully and reflectively, you can actually use some of the pattern recognition that computation data processing is really good at to reveal some of these systemic inequalities that you couldn't see by just looking at the data set by hand.

This offers students a crucial perspective to approach their work more critically as designers of computational systems in society. On the whole, P5's critical pedagogy looks to empower students to define what it means to be a "good computer scientist." By introducing students to sociology in their CS course and making that a central theme of the course, (not merely a one-off lesson on the social impact of computing), P5 provides students with an alternative way to understand and practice computer science. As they tell their students at the beginning of the course:

> This isn't me as a social scientist trying to invade technical spaces, this is me asking you as a CS student to reimagine what it means to be a programmer, so rather than thinking "ok I know how to program and I can apply some of these writings to my own work," it becomes, "what if being a programmer, or being a good programmer, meant to both know python but also know these readings, that actually it's part of this same core expertise?"

P5 is explicitly inviting students to imagine an alternative vision of computer science by empowering students with knowledge of both sociology and computer science. With students understanding the limits of computing and the social impact of computing, they

are enabled to chart a new computer science identity, one which includes being critical of computational systems and their uses in society.

3.4 Discussion

We finish the chapter by discussing the vignettes in more detail. First, we describe selecting the vignettes. Then, analyze the vignettes using our theoretical framing which brings forward how hegemony appears in several places in the experiences of our participants. We then discuss the limitations of this aspect of our study. Some parts of this section, in particular sections 3.4.1 and 3.4.3, similarly draw on a forthcoming paper I wrote on this aspect of the data named *Critical Pedagogy in Practice in the Computing Classroom*.

3.4.1 Selecting vignettes

Picking vignettes to illustrate the wider picture given to us by our 13 participants was a challenge, as our data is rich with fascinating vignettes of critical CS pedagogy. Given this research was initially for a conference with a short page limit, we were not able to include all the vignettes we would have liked to. Here is a quick summary of some teaching examples we would found noteworthy but could not include:

- An example of a participant who listens to students' anxieties and unpacks them in a way that helps students understand the unfair structures they operate under.
- An after-school program for primary and secondary students centered on teaching students about technology in order to further student activism. Using media, videogames, and augmented-reality apps, students use this technology to bring forward issues they face in their lives. For example, students create videogames with mechanics that bring forward the difficulty in accessing the only water fountain in their school not supplied by lead pipes. In another example, students create an augmented reality app that helps community members explore the history of their

historically dispossessed community, such as where famous athletes from their community first learned their sport.

- Teaching that includes opportunities for students to understand the structure of the university, in a bid to support student activism in changing the university structure to improve fairness, e.g. admissions and funding.

Our participants described practices that fit with a number of published experience reports of critical pedagogy in CS: at the intersection of machine learning (ML) and CP, Yim Register's research demonstrates how self-advocacy can be infused in ML pedagogy. By training ML models on students' own data retrieved from Facebook, students were better able to articulate criticism of ML models [100]. Research by Lee & Soep articulates a pedagogical framework that enables students to "create and disseminate digital projects" that "challenge unjust systems" [67]. These studies (and more) follow similar objectives and trajectories as the vignettes listed above.

3.4.2 Hegemony revealed from vignettes

Analyzing these results using our theoretical framing, we see hegemony appearing in all three interviews, albeit in different ways. For instance, P3's teaching not only resists the God Trick, but we also see how hegemony affects students' expectations of their CS class. P5's vignette reveals a careful pedagogy that resists hegemonic norms like rendering technical. Finally, P12's vignette is clearly entangled in the constructionist hegemony, while at the same time including important ways rendering technical are resisted. Each participant's vignette is unique in how hegemony appears in these examples.

Participant 3

Consider P3, who centers teaching knowledge about the "hyper localized community" of the students. This pushes against the God Trick described by Haraway by narrowing in on the context of students and their local history. This directly counters "universal"

knowledge that "sees everything from nowhere" [58]. Instead, P3's teaching resists the arbitrary distinction of "technical" and "social" that permeates CS education by centering on both programming and local history in their teaching.

An interesting result of this teaching is how students respond. Consider how students were confused they weren't using the computer in P3's CS class, questioning "why are we learning all this stuff" and not just using computers. Students' confusion and questioning signal how they implicitly expect their CS class to focus solely on technology and free computer use. Using our framework, this reaction is a result of the hegemonic norms of constructionism and rendering technical in CS education, namely that to learn CS is to be coding on the computer, not learning this social "stuff". This reveals an interesting way students are inclined to react to CP. Predicting and mitigating this reaction is valuable knowledge for teachers looking to use CP going forward.

Participant 5

Out of all our interviews, P5's teaching stands out as a well-thought-out approach to resisting rendering technical in their CS teaching. P5's vignette, where socio-political readings and discussion happen alongside introductory CS classes, directly connects the "social" and "technical" aspects of technology. This resists the norm of rendering technical by showing students how sociopolitical knowledge and programming knowledge go hand in hand. In particular, P5 directly challenges the identity that enables rendering technical with explicit conversations about reimagining what it means to be a "good" computer scientist to include the sociopolitical knowledge of CS. Another strong aspect of P5's vignette is how this framing happens at the beginning of students' study of CS. This helps address the norms students may adopt early on as beginner CS students, and go on to alter the students' vision of CS before they take many more classes in their degree, which can cement the "technical/social" divide.

Participant 12

P12's vignette shows a deeper entanglement with constructionist and liberal hegemony in CS education. As P12 shares, they explicitly identify with constructionism and see an important link between CP and constructionism. This is evident in their teaching too. They describe their teaching around bringing technology into the classroom and providing students with free access to these materials to do as they wish. Students then are expected to use these tools to play and explore through their own passion and volition. This play doubles as a vehicle for learning about technology. The emphasis on free, unstructured time with technology is deeply tied to the liberal ideals of individualism, freedom, and autonomy, as well as in line with a constructionist pedagogy.

The use of constructionism in teaching CS poses a serious threat to the potential CP has at changing CS more generally. Suppose the workshop P12 describes had students who aren't socialized to play with technology or see themselves as technologists, such as young girls, who are more inclined to use the freedom offered by constructionism to play with traditionally girl-focused materials (like dolls). In placing such a high value on student autonomy, pedagogies like constructionism potentially run the risk of hardly changing the way marginalized students see and interact with computing. This only serves to reinforce the gendered and racial norms that maintain CS as a discipline primarily for white and Asian men. However, emerging research on constructionism in the last 10 years is beginning to address the traditionally apoliticized aspects of constructionist pedagogies [18].

That being said, this example also resists some hegemonic norms in CS personhood worthy of discussion. Consider the techno-solution mindset in hegemonic CS identity: developers, already belonging to the dominant culture of CS, are equipped to create technology to solve inequalities of those often outside of tech. In P12's teaching, they instead empower the students themselves to develop context-sensitive technology to address the needs of their community. This itself is an important way the techno-solutionism of CS

hegemonic identity is resisted by positing CS as a tool students can leverage to address their own wider struggles.

Furthermore, allowing the students to create resources to help them do "illegal" things (like accessing basic necessities of modern life) is itself quite counter-hegemonic. Following laws, especially cruel and unjust laws are textbook hegemony. That P12 allowed students to support "illegal" activity like taking access to electricity to use kitchen appliances is an important insight from this vignette and an action any CP practitioner looking to resist hegemony must consider.

3.4.3 Limitations

It is worth noting that all of the chosen vignettes come from participants who are men. Indeed, an observation from our interview is the way men and women conduct CP. More often than not, men participants from our data described starting the term off with a radical reconfiguration to set that tone from the start. Whereas our women participants described starting on a more traditional tone and guiding students into CP over the course of the term.

We suspect that this is because men have more leeway in the way they teach. It is well documented that women are unjustifiably rated worse on course evaluation compared to their men counterparts [20]. For this reason, it is sensible that women would opt to transition the class to a more student-centered approach to avoid negative reactions when mixing up student expectations.

However, one factor that went into selecting vignettes is how striking they were. This may help explain why men were chosen for each vignette: since they are less scrutinized by students they are able to take more risks, resulting in flashier approaches to CP. Future work should consider the gender aspect of CP and explore how gender affects CP.

Similarly, another limitation of this study is it only selects a small sample from our data. We interviewed 13 participants but only highlight three here, as well as analyzed the underlying hegemony in these vignettes. Given the scope of a Master's thesis, analyzing

all interviews individually would be too much work. Instead, in the next chapter, we synthesize the beliefs, practices, and attitudes of our participants into a cohesive whole. From there, we will analyze the hegemony in this larger summary of all participants.

Chapter 4

What is Critical Pedagogy? Creating an ontology from practitioner description

Although the previous section provides a rich description of CP use in the classroom, it remains unclear what CP even is. CP has been criticized by many researchers (chiefly by Apple [9–11] and others [30, 56, 112]) for being overly theoretical. Take Tower's keynote address *Reinventing the university: Visions and hallucinations*, which "outlines a range of literature [on CP ... which] offer little in terms of concrete implementation strategies" [81]. Indeed, this lack of concrete strategies is part of what motivated me to start this study in the first place!

This chapter seeks to address this critique by providing an *ontology* of CS that emerged from our interviews. By ontology, we mean a categorization of the properties and concepts our participants described CP as, and how these categories relate to one another to create a cohesive whole [115]. Defining an ontology is a common practice in CS, as software often models real things. So much so that the dominant paradigm in programming is called *object-oriented*, which is designed to facilitate modeling things as they are using objects as the main abstraction (hence "object-oriented"). Although we did not initiate this study seeking an ontology and instead only were looking for concrete practices of CP, our training in CS led us to notice an ontology merge from our interviews.

We define CP using an ontology, as there is no strict ordering in the underlying concepts. We chose this as opposed to a taxonomy, which is like an ontology with a hierarchical organization, which we found unnecessary. Indeed, in the components of CP, we found no strict ordering that educators must follow to do CP, instead, educators can start from any direction.

All participants are represented in one way or another in this ontology. Of course, not all participants will agree with every concept laid out in this section. However, our effort is to identify concepts that many participants bring forward in our interviews. Indeed, this section will center and categorize the most common ideas that came out of the interviews. Consequently, the organization of the commonalities gravitated towards four planks: (1) meeting students where they're at, (2) rethinking the student-teacher relationship, (3) talking about power in the classroom, and (4) taking action. To illustrate these planks and their components more easily, we provide an illustration that outlines the planks and concepts that undergird our ontology of CP. Each plank is elaborated on in its own section, with subsections that further detail each plank.

4.1 Plank 1: Engaging with students on their terms

The first plank of our analysis coincides with a practice that participants report starting early and continuing throughout their teaching: meeting students where they are. In broad strokes, this means seeking to understand students and their particular context. Understanding students is far-reaching in CP as it allows for two other necessary parts of CP. Firstly: it allows the teacher to build trust with their students. Trust is an essential part of CP, one that comes up often throughout the interviews. Secondly, it enables a constructivist pedagogy by equipping educators with the knowledge necessary to make their teaching responsive to their students.

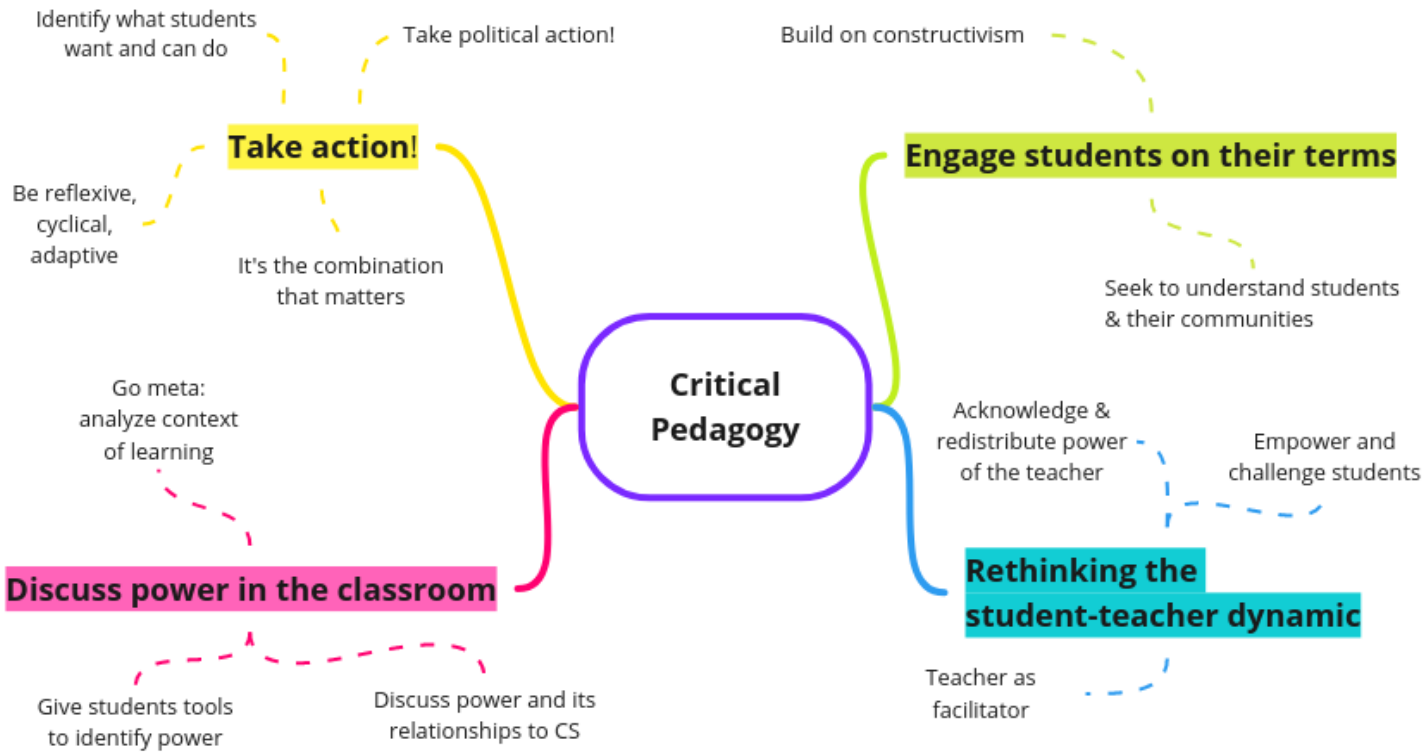

Figure 4.1: Diagram of our ontology of CP. The four planks attach directly to CP, and each plank is surrounded by its sub-components.

4.1.1 Seek to understand students and their communities

All of our participants believe understanding students and their context is the utmost priority. As P10 describes, teachers must "do their best" to learn about students' identities, fears, aspirations, and what students care about. This process is symbiotic with building trust. One participant, P3, highlights how committed they are to building trust with students that they report prioritizing building trust over correcting some students' mistakes at the beginning of a new class:

> [when students first start learning] there might be misunderstandings but I don't jump in to correct them right away. I would even say, on a semester level, I definitely let a lot more go during the first couple weeks, even if I know it's inaccurate, with the intention of building community and trust in

> the community, between me and students. If I shut them down, no matter how kindly I do it, it already changes the dynamic and gives the impression: me expert, you novice.

This teaching practice deliberately trades correcting students in order to prioritize building trust in the classroom.

Furthermore, it is the responsibility and duty of the critical pedagogue to understand students, including about their cultures and identity. It is not the responsibility of students to educate teachers; instead, this is the teachers' job to learn about their students. To this end, our participants report talking to students about their interests and what students wish to learn at the start of the course, sometimes using surveys.

P3's approach is noteworthy in that they frame the process of learning about their students as "being a cultural anthropologist" of the students and their communities. This includes doing things like observing students outside of class time to get a better sense of the student's lives. This participant also reports intentionally interacting with students outside of class time. This is particularly effective as it reveals to P3 the communities or groups the student is a part of. Indeed, learning about students' social context can help build knowledge about students. This is why some participants, like P1 and P7, build relationships with the students' parents, families, or peers.

Building these relationships with students establishes trust. Trust is particularly important to CP because it allows students to be vulnerable during the learning process with their teacher. As P9 describes, "learning is a very emotional process". As we will see in other chapters, learning through CP can be especially emotional. Feelings of discomfort, confusion, or even distress are, at times, a consequence of CP, given how CP involves unearthing uncomfortable realities about society. P9 elaborates how "developing a trust with your students [allows them to] be vulnerable and open to learning". In other words, trust helps students face these unpleasant emotions with the support of their teacher.

4.1.2 Build on a constructivist pedagogy

This emphasis on trust dovetails nicely with the emphasis many participants placed on constructivism in CP. Many participants directly name constructivism as highly influential to their CP, with some noting constructivism underlies CP. Others describe their teaching as constructivist in all but name, by elaborating how students have pre-existing knowledge walking into the class, and how important it is to leverage this pre-existing knowledge when helping students build their understandings of new concepts.

An important and, at times, challenging part of constructivism in a CS context is contextualizing the "technical" knowledge that will be presented to students in class. Contextualizing knowledge, especially scientific or computational knowledge, is an important part of many participants' pedagogy. In a CS class in particular, participants (like P10, P5, and P7) encourage students to "take an epistemic line of inquiry" when learning scientific facts, including discussing the history of science and CS. For P11, this means "widening the scope" of what is normally discussed in the technical class, such as the social and historical aspects of computing. P10 echos how the norm in CS is to present concepts as "decontextualized" facts:

> The big issue is you look at machine learning pedagogy (if you can even call it that) [is we] teach through like closed form mathematical, you know, formulas — like if you look at the famous Coursera Andrew Ng course, it's like, here's all the thetas.

Although our participants identify this is usually how CS knowledge is portrayed, many participants describe this kind of teaching as antithetical to CP. As P5 describes: "uncritical pedagogy is non-reflective pedagogy [...] any pedagogy that doesn't take into account issues of society, power, identity, and subject formation is uncritical." Instead, a CP is aligned with constructivism in that, to best support student construction of knowledge, educators must present knowledge with rich connections to historical and social aspects as to better support student comprehension.

4.2 Plank 2: Rethinking the student-teacher dynamic

The second plank of CP that emerges from our analysis is the student-teacher relationship. All of our participants, in one way or another, share their commitment to redistributing the power in their classrooms. Sharing the power of the teacher with the students changes the role of both the teacher and student in significant ways. This section will elaborate on the new role of students and teachers under CP.

4.2.1 Acknowledge and redistribute the power of the teacher

Although our participants conceded that a power disparity between the student and teacher is unavoidable, CP nonetheless demands scrutiny and action to minimize the power imbalance as much as possible. As P9 remarks, it is hard to do CP because the power imbalance between student and teacher is "too great" to fully balance; however, it is imperative to still "attempt to". Our participants report doing this in many ways, such as being mindful of their biases and limitations and providing students with some power over the content and direction class.

Even though being the teacher comes with unavoidable power, critical pedagogues must reflect seriously on how to use this power "ethically and responsibly" in the words of P6. For starters, being aware of one's biases and blind spots is fosters a more just classroom. Participants offered an assortment of practices and cues for reflection for teachers to, in the words of P10, "check yourself":

- Consider when people raise their hand, who do you normally call on? (P4)
- Does your teaching allow students to engage prior knowledge?
- Does your teaching give opportunities for students to share how they want the course to unfold or criticism of your teaching?
- Does your class allow room for students to learn in ways meaningful to them?

Of critical importance to sharing power with students is relinquishing the idea the teacher is the "sole knower" in the classroom. Our participants, like Freire, are deeply critical of the idea students are "empty bank accounts" waiting to have knowledge kindly deposited into their heads by teachers. Instead, teachers should "[be] open and reflective about what types of knowledge and subjects you are permitting to exist in your classroom and which you are sidelining" (P5). Bringing in students' prior knowledge and interests in the learning process represent an important way teachers can empower students in their class.

4.2.2 Teacher as facilitator

The characteristic that participants often used to describe the role of the teacher under CP is that of "negotiator" and "facilitator". This job starts with the previous plank, learning about student interests. From there, teachers need to share the power of their class to balance the interests of students alongside the required curriculum. Our participants also remind us the teacher is a stakeholder too, with their own personal objectives that need to be similarly considered.

Bringing students into the decision-making process of the course is one way our participants describe negotiating student needs. Many participants report adjusting course materials throughout the course to match student interests and feedback. P7's approach is interesting in they intentionally avoid planning the latter half of their course, and instead allow students to decide what is covered. Nearly all of our participants describe using open-ended projects to empower students, including giving students power over the design and direction of the project.

Another aspect to be balanced by the teacher-as-negotiator framing is the interests and mandates of the teacher themselves. Our participants report it is not in line with CP for the teacher to say "learn whatever you want, I don't care" (P12). Instead, teachers need to balance the students' interests with their own, as well as the material they're mandated

to teach. However, doing so without falling into an authoritarian teaching style requires tact. P12 illustrates how they do this in their teaching:

> [Teachers are] entitled to having your own ideas, and views on things, and even to bring your own topics and say, "Hey, I'm here to propose a discussion or propose that we learn something." [Sometimes,] students have to take a leap of faith and give you the opportunity of introducing something that they might not even know exists, right, and they might not be interested because they just don't know the thing even exists.

Here again, we see the importance of trust in establishing a classroom environment that allows for the authentic sharing of power and interests in the class. This quote also shows the way a teacher can carefully initiate a dialog about their interests too and begin to balance the interests of the class between student and teacher.

4.2.3 Empower and challenge students

On the other hand, our participants also described at length the role of students in CP, including what responsibilities students have in CP that teachers need to enable students to adopt. Under CP, students question the content they're learning and engage in critique (including critiquing the institutions they find themselves in). Our participants assert students must take responsibility for their learning in CP, and play an active part in the learning process as opposed to a passive role.

Of course, given the inherent power of the teacher, our participants are quick to point out teachers play a crucial role in fostering an environment conducive for students to take up this role. Firstly, a safe learning environment is essential to foster student ownership of their learning. To make a safe learning environment, our participants frequently mentioned some practices to this end. The first is the aforementioned plank, building a trusting relationship with students. A trusting relationship between student and teacher

builds a culture of trust in the classroom, thereby allowing students to be vulnerable and take risks in the classroom.

Our participants report the vulnerability that follows from trust affords students the space to reflect and discuss the challenges they may face when learning through CP. As mentioned before, CP can illicit upsetting emotions as students come to terms with their oppression or review their assumptions about meritocracy in society. To support students as they process their emotions, several participants (such as P5's intro to programming class or P6's software engineering class) structure their classes around weekly readings and small group discussions with guiding discussion prompts. P11 finds this approach useful as students can share their emotions and report feeling not alone in this challenge.

Finally, the process of taking ownership of learning is a delicate one, and participants must deliberate on how to best enable students to take up this role. P9 describes how taking ownership of learning is a "delicate process" as students are not accustomed to this way of learning. For that reason, shifting responsibility for learning to the student must be careful and gradual. Ultimately, the role of the student under CP requires work from the teacher to facilitate the uptake of this role.

4.3 Plank 3: Discuss power in the classroom

As elaborated on in the previous section, considering how a teacher manages the power given to them in the classroom is an important aspect of CP, but this isn't the only important way power manifests in CP. Under a CP framing, power must also be explicitly centered and named in the teaching of any discipline. This section outlines how our participants center power in their content as well as how they help students foster the skill necessary to identify and question power structures in society.

4.3.1 Give students the tools to identify power

A common element of teaching social power according to the participants emphasized providing students the tools to name and question power and how it operates in students' day-to-day lives. In the words of P7, CP is about making "implicit power structures explicit" by naming the ways power operates in covert ways. P10 remarks how computing is a site rich with covert power, in particular, how "hard to understand" computer systems often mask the power relations embedded in technology, and how programmers play an integral part in designing these systems. Exposing students to the power operates is no small feat, as P8 articulates power is very often "disguised", "masked", and works to "divert" attention away from itself. However, building off the previous planks, teaching power in a way that is connected to students' experiences can provide a fruitful way to open students' consciousness to these otherwise invisible social forces.

The following anecdote from P6 illustrates how power is hidden from students and strategies to help students see the subtle ways power operates in their lives. P6 teaches a software engineering class, and conversations with students reveal a "central anxiety" students have going into this 4th-year software class: students worry about finding their first job. This is an understandable anxiety for a graduating student, however, P6 shares how these students have limited understandings of "what a labour market looks like, or how recruiting works, or any of these structures that kind of shape the space of opportunities that [students] have access too".

To address students' worries and expose social power to students, P6 uses a Socratic method to debug students' understanding of employment. They start by asking students, "how do you think [hiring staff] are deciding [which applicants get an interview]?" For some students, this is their first realization "oh, yeah, there's a person deciding." Then, P6 gets students to consider, what information on their resume is used for hiring. Some students claim grades are an important factor in judging candidates, however, P6 points out to students they haven't put their grades on their resumes.

For many students, this dialogue gets students to examine their beliefs about hiring and leads to students "realizing that hiring is this really, really subjective, murky process that's based on bad information". This is because, according to P6, "most of the people who are pursuing those jobs don't do a good job conveying [important information about their candidacy] because they don't know anything about how people are perceiving it". This lack of clarity or standardization in resumes then clarify to students why word of mouth is such an effective way of getting a job, and the injustice surrounding this norm. Ultimately, P6 feels they've succeeded when students "leave the class feeling like they understand how broken things are and how they have to navigate it". This story illustrates how educators doing CP can leverage students' experiences to elucidate how power works for students.

4.3.2 Discuss power and its relationship to CS

Another approach to centering power in the class content is focusing on the relationship between power and the content at hand. In this context, participants discuss how power and computer science are related. This is not usually how CS is framed, but as P12 remarks, everything is related to power: "no matter what topic you're looking at, those kinds of societal issues are always fair game" and eligible for discussion in class. Indeed, there is always a way to connect content to power.

Several participants, like P7, find it particularly important that students understand the connection between power and CS as technology becomes a strong social and political force in society. In a sobering example, P7 elaborates why it's so important to consider power and politics in technical decision-making, as it could have serious, life-changing impacts on people:

> These are real people's lives, when you become a data scientist, you could be participating in creating a model that takes someone's children away, you could be participating in creating a model that keeps someone in jail for the rest of their lives, like, hold on a minute, right?

This showcases how technologists have a great responsibility to consider the impacts of the technological products they create.

In a different framing, P13 emphasizes technology is a growing requirement for anyone to engage with politics, not just programmers. At the time of writing *Pedagogy of the Oppressed*, Freire frames literacy as a necessary requirement for citizens to participate in politics. However, that was roughly 50 years ago, at a time when Freire could not have foreseen how important technology would grow to become in our contemporary time. Given the pervasiveness of media and technology in contemporary society and politics, P13 extends literacy to media literacy as a requirement to engage politically in society, on par with reading and writing.

4.3.3 Go meta: critically analyze the context of learning

However, our participants report it's not enough to abstractly analyze power operates in society more widely or in CS specifically. Instead, our participants emphasize turning the gaze of identifying and scrutinizing structures of power onto their own teaching and educational institution. No one is outside the text, and the same goes for instructors looking to expose power relations more generally. Take for instance P7, which encourages students to question their authority as a white educator when discussing issues of race. Another example comes from P5, who starts their programming courses with the theme: "Your education is never neutral". In this lecture, the participant details how political forces have shaped and continue to shape the university and CS curriculum. As P7 reminds us, "you can't talk about the system without acknowledging your place in it." This means being honest about the power imbalance between student and teacher and for P9 and P11, confronting how education is largely run by neo-liberal values:

> the real purpose of education is not so much to teach anything, it's more or less to make sure a modicum of socialization is achieved, and so social order is preserved [...] by making obedient workers (P11)

Obviously, our participants look to reverse this process to the best of their ability, however hard that may be. This is in part because critical pedagogues hold that education and schooling, in general, are redeemable and possible to reform, a view that differentiates CP from anarchist pedagogies.

4.4 Plank 4: Take action

This brings us to our final plank of CP: taking action. It is not simply sufficient to criticize these systems of education or CS more generally. Instead, our participants are adamant a central tenant of CP is taking some sort of political action to actualize a better world. This section will outline how our participants do this.

4.4.1 Identify what students want and can do

Making political change must originate from the students and their ideas, according to our participants. The point of helping students identify power structures is so that they can "[move] beyond the "here and now" and envision an alternate future" (P12). However, the ultimate decision of what issues students want to get involved in changing is to be made by the students, not the teacher.

An important job for the teacher then is to help students see what they *can* reasonably change. The role of the teacher according to P9 is to help students see not only what is "possible" to change, but what is "easy" to change. When P9 uses "easy", they are not referring to trivial changes, but instead changes that maximize the effort to affect ratio. For instance, reversing the entire course of capitalism is aspirational, but likely less fruitful than simpler changes, local to students. This is important because as P6 points out, students are new to thinking about society in this way. They are not familiar with how these intuitions work, and that unfamiliarity serves as a hindrance to effective political change as students don't know where to target their efforts.

Some participants, like P5, 8, 10, and 12, include activities of imagination to support students' process of identifying what they want to change in order to actualize a better world. For instance, after teaching students about the environmental and social harms of resource extraction for computing, P8 asks students to imagine "what would a non-extractive economy look like?". For P5, after exposing them to sociological readings that expose power and issues of injustice in technology, asks them to "re-imagine what it means to be a [programmer] [...] what if being a good programmer meant to both know python, [but also knowledge of social and political aspects of computing]?". As P12 shares, it allows students to put new ideas into the real world. Imagination can provide a powerful tool to help students tap into the injustices they want to have resolved in their conceptions of the future.

4.4.2 Take political action

Many of our participants were harshly critical of any kind of CP that doesn't involve making some sort of change. Herein lies a unique characteristic of CP and CS: since CS currently plays such a powerful role in society, it can be leveraged as an instrument to enact change. Indeed, learning these technologies can enable students to resist the harmful consequences of technology, as P12 describes:

> So when you say, "Okay, Facebook is terrible," or "this app is bad," or "I wish there was an answer to this," [then] you actually have to engage with designing something, and learning how to program and learning the technical side of things [to offer a solution].

Similarly, P1 emphasizes how they expose students to technological tools for the express purpose of "implement[ing] and [to] do the activism that is at the core of what we do". For P1, the goal is to "increase civic engagement of young folks" through learning about technology. Simply put, critique isn't enough, and action must be taken to adjust inequal-

ities when students learn to identify them, and CS knowledge can be of service in this effort.

Indeed, some of our participants saw the potential for CP to focus too much on criticism and not enact solutions to address the harms of technology. Some worry that being critical of technology is becoming a "spectator sport" and is currently the "new fashion" (P12). For P8, they saw the overly critical risk of CP results from CP's close relationship to Marxism, with P8 remarking they "don't want to wait around for a revolution". This worry that CP is "waiting" for a revolution exposes the risk that CP does not adequately address the need for computing education to result in a meaningful change on a societal level.

Some participants, like P8, 12, and 13, name constructionism as a complementary pedagogy that addresses the potential for CP to be without taking action. Given the importance constructionism places on doing and creating, the idea is this pedagogy is well suited to ensure change does happen. Indeed, a subset of our participants saw constructionism and CP as deeply related and important that they go together.

Others believed that organizing is a central part of CP, despite the barriers participants described in doing so. For instance, take P6, who identified organizing as a central tenant of CP, but confesses they've never directly facilitated organization with students. Instead, participants like P8, P9, and P10 encourage students to organize. P1, who works in the informal education space, is the only participant who describes how their teaching involves organizing, such as mobilizing students in protests and marches. Although activism is central in many participants' conception of CP, participants report limited engagement with activism on this front.

4.4.3 Be reflexive, cyclical, adaptive

In the final suggestions to teachers looking to do CP, a crucial part of CP is being reflexive, cyclical, and adaptive. Criticality is a journey, not a destination, and practitioners of CP need to remember to continue to iterate and improve on their teaching. Similarly, tech-

nology and politics are always evolving, so a teacher looking to do CP must remember to continually look to re-examine their beliefs and teaching, making adjustments along the way.

This means doing continual reflection where teachers assess their teaching methods. As P11 puts it so succinctly: "what kind of society are we building with our teaching?". This question must guide the teacher in the continual discourse with themselves in adapting their teaching content and delivery, as well as epistemic beliefs. Throughout the interview, many prompts for reflection are offered: what worked and what didn't? How can you improve your teaching next time to reach more students? Constant reflection helps guide the CP-inspired teacher through the "process" (P7) of being a critical pedagogue. Constant reflection also helps the teacher be clear to students what the teacher's goals are for the class and justify their choices.

4.4.4 It's the combination that matters

Finally, it's all of the above that matter in doing CP. All of the above facets of CP must come together to create a cohesive pedagogy aligned with CP. Indeed, many participants, like P8, 9, 11, and 12, all assert in their own ways that "there is no one trick" to doing CP. Instead, P11 shares how CP is an identity, not a collection of tricks and premeditated tactics. Instead, it's all of the above characteristics that must be developed to make a coherent pedagogy.

4.5 Discussion

This section, like the previous chapter's discussion section, will discuss how each facet of CP relates to hegemony. Our theoretical framework, including concepts like rendering technical, liberalism, and feminist epistemology all reveal themselves in the planks listed above. We then go on to discuss limitations in regard to this part of the study.

4.5.1 Hegemony in the four planks

Indeed, our participants' conception of CP interacts with the hegemony used in our theoretical framing. At times, hegemony is resisted, while at other times, participants' conception of CP aids hegemonic norms like liberalism and rendering technical. Each plank will be analyzed in relation to the above hegemony. Given the size of the interview data, this is a non-exhaustive list of the ways hegemony manifests in CP.

Plank 1

In the first plank, we see efforts in our participants to resist the God Trick by meeting students where they are at. Consider P3's approach to building trust with the students: instead of correcting everything students say at the beginning of a class, they "let a lot more go" at the beginning of the semester. This is opposed to the teacher asserting the "objective" "universal truths" as the only permissible knowledge in the classroom. Instead, students are afforded the space to articulate their partial understanding of content as they construct their knowledge.

Participants' efforts to contextualize CS knowledge also represent an important pushback to rendering technical. As participants like P6 note, they actively resist the decontextualized and overly "mathy" framing of CS knowledge, common in CS education broadly. Challenges articulated by participants, like the difficulty contextualizing CS knowledge in Section 4.1.2, show the deliberate effort participants make to contextualize CS knowledge despite the difficulty given the dominant framing of CS. Indeed, P11's claim that CP in CS must "widen the scope" of what is considered "real" CS encapsulates well the ways CP resists the hegemonic norm of rendering technical, by dispelling the notion only technical knowledge is considered worthy and appropriate for consideration in a CS class.

Furthermore, by encouraging an "epistemic line of inquiry", participants are also equipping students with the skill and knowledge to resist the God Trick. Finally, by highlighting to students the historical and social-political contexts of CS knowledge produc-

tion, participants reveal the personality and partiality of knowledge. This helps remind students that even "technical" or "scientific" knowledge is still impacted by subjectivity.

Plank 2

The second plank, which reconfigures the student-teacher dynamic in the classroom, resists objective and authoritative claims to knowledge. By rethinking the student-teacher dynamic to center students' perspectives and usurp the teacher as the "sole knower", participants actively disrupt the dominant framing of scientific knowledge, which relies on authority to render "universal", and "objective" facts about the world. Instead, highlighting the important and personal perspectives students have, like their interests and prior knowledge, helps provide a more nuanced conception of knowledge. It further resists the tokenization of knowledge into discrete units of "truth" that education systems reify with evenly packaged, credited courses that eventually get cashed in for a degree. Finally, participants also use reflection about what knowledge the teacher allows "to exist" in the classroom to stay continually resistant to the hegemony of "objective" knowledge that teachers hand down to students.

Plank 3

Through plank 3, we see how our participants' conception of CP similarly responds and resists hegemonic forces like rendering technical. By discussing explicitly the power that technologies have, our participants are directly resisting a framing of CS knowledge that disconnects its products from its social consequences. Recall P7's quote where they explicitly call out how technology can have serious consequences on "real people", such as algorithms that determine prison sentencing or parents' custody rights. This way of teaching explicitly seeks to make clear to students how technology has important impacts on society. Similarly, remarks by P12 that assert there is a way to discuss social issues in all domains (even "technical" or scientific ones) clearly mark how our participants are

seeking to resist the dominant hegemony that posits "technical" and "social" knowledge as separate entities.

Furthermore, our participants are acutely aware of how education is mobilized as a tool to socialize "obedient workers". Instead of using their teaching to produce workers, participants like P13 liken technology to reading and writing in terms of how they enable students to be politically aware and active. This shows how CP is seeking to dismantle the "technical/social" divide and advance the ways that technology can support activism and political engagement.

Similarly, P6's discussion of helping students see the brokenness of the hiring process helps shake liberal, hegemonic beliefs about meritocracy. In this example, P6 shares how they get students to see that hiring processes are not an objective assessment of students' skills but a "really really murky process" that perpetuates inequalities. This directly pushes against exalted liberal values like meritocracy by exposing that there does not exist an objective standard by which employers judge applicants. In its place, students realize that the hiring process is a subjective process that relies on patterns of privilege (namely, how people's social capital and network more often land them a job — not some objective judgment of their skill). This also points to the hegemonic norms students adopt around meritocracy and liberalism in the labour market.

Plank 4

Finally, our last plank on challenging power structures understandably reveals many instances of hegemony. Given this plank seeks to address power and inequality directly, hegemony is bound to emerge in these discussions. The first instance of hegemony we highlight is how students are largely unaware of what they want to change (or, for that matter, can change) about society. This speaks to a hegemonic norm for students in CS that disconnects them from the sociological perspective of society and how they can make a difference.

At the same time, the way participants discuss helping students take action ties into hegemonic liberal norms of individuality and change. Here, the focus is on getting individual students to make systemic societal changes. Indeed, P12's insistence that criticizing technology must come with a solution seems to fall in line with the myth individuals are somehow responsible or capable of altering systemic issues. The notion that students should endeavor to design some technological solution to address the bloated power of companies like Facebook fit well into rendering technical.

As our participants point out, however, this is not to suggest that the counter-hegemonic approach to political change is "waiting around" for a Marxist revolution. This itself represents yet another hegemony our participants identify and seek to avoid. Instead, our participants seek to disrupt oppressive structures instead of simply waiting around for a change — for instance, P6 centers on the importance of student organization and action as critical tenants of CP.

At the same time, participants describe limitations in supporting students in this organization, citing institutional barriers as disabling them from answering this call for organization. Instead, some participants (like P6 and P9) "encourage" students to take their own actions. This approach is worth scrutinizing in terms of its effectiveness and the way it upholds hegemony. For instance, this distinction that individual students should be the ones doing the organizing and disrupting (not alongside professors/teachers) fits a hegemonic norm that benefits those in power by disrupting relationships (e.g., teacher-student) that could result in effective social change.

Our last observation on hegemony in the CP described in this chapter spotlights how participants describe being reflexive and adaptive as a way to resist hegemony. Indeed, by carefully reexamining beliefs and teaching practices and asking oneself "who does this benefit?" our participants stay on their toes in identifying hegemonic practices and ways of thinking. This facet of CP is a meaningful way our participants can continue to do the tricky and hard work of resisting hegemony. However, several participants note that

CP is not "a trick," and similarly, identifying hegemony is not a one-and-done process. Instead, it must be actively reflected upon to resist effectively.

4.5.2 Limitations

Limitations of this study come from the sample of our participants. Initially, this study was initiated to illustrate practice applications of CP in the CS classroom. This is to satisfy my curiosity about what CP *actually* looks like in practice teaching CS. However, from our data, we notice this more extensive ontology of CP. Herein lies the main limitation of this aspect of our study: we only interviewed *CS* teachers who use CP. Indeed, talking to critical pedagogues in other domains like math, science, language, and arts (to name a few) will likely yield differing results.

Another limitation comes from my partial perspective as a researcher. On the one hand, I have been more generally acquainted with the hegemonic forces of CS and education. I, too, am undoubtedly influenced by these hegemonic forces and have blind spots of my own. On the other hand, I feel distant from my participants in a few ways. I'm much younger than the average age of my participants and early on in my academic and teaching journey. For this reason, my analysis could be more radical than my participants.

Finally, the last limitation of this ontology is how useful it is. Is the above ontology helpful in guiding aspiring CP educators? Or is it too ambitious or unrealistic? More research in this area is needed to answer these questions. Indeed, I'm unsure how I measure up to this ontology. In some ways, I see my teaching falling in line with this research, but just as I critique my participants for lacking concrete action and organizing, I have yet to do this in my teaching. For this reason, our ontology includes being cyclical and adaptive to give ourselves leeway and flexibility in implementing CP in the classroom.

Chapter 5

Discussion

5.1 The many shades of CP

The two preceding results chapters provide very contrasting ideas of CP. Chapter 3 provides a rich description of CP using the three vignettes selected from our interviews. The intention here is to highlight how teachers adopt divergent perspectives on CP, and especially how context specific these adoptions are. Our results show that doing CP in impoverished neighbourhoods of Brazil (P12) will look different from doing CP in historically dispossessed neighbourhoods in the US (P3), which again is very different from doing CP in the US university context (P5). Indeed, this is perfectly normal and expected of a pedagogy that seeks to meet students where they are and address their local context. Despite the differences, these vignettes still push back against rendering technical, where complex social issues are rendered into simple technological problems with simple technical solutions. However, it is still possible to distill the overlap in the instances of CP showcased in Results 2. Our data contains a non-negligible amount of commonality among the values, beliefs, attitudes, and expectations among teaching professionals doing CP. This is the goal of Chapter 4, to bring forward that commonality in an attempt to articulate (in general terms) how CP is conceived of in practitioners' minds.

To that end, our results address our research questions outlined in Section 1.5. Our first research question, “for CS educators who are inspired by CP, what instructional interventions do they describe using when teaching CS critically?” is described in both Results chapters. Chapter 3 provides an in-depth illustration of the teaching practices our participants describe carrying out in their CP class. Furthermore, Chapter 4 provides more general examples of our participants’ teaching practices. Our second research question, "For CS educators who are inspired by CP, what elements, concepts, and considerations do they describe as constituting CP?” is largely addressed in Chapter 4. This chapter synthesizes the ideas that emerged from our interview data, such:

- Understanding students’ contexts and interests;
- Incorporating students’ perspective in the teaching interventions;
- Building on constructivism;
- Rethinking the student-teacher dynamic to share power and facilitate student learning;
- Discussing power in the classroom to empower students;
- Teaching in such a way that enables students to take action and challenge unjust power structures in the education institution and beyond;

These ideas emerge consistently throughout our interviews and represent the conceptual components that our participants articulate as “doing CP”.

5.2 Hegemonic forces in critical CS pedagogy

In the two discussion sections that follow the results of the interviews in the previous two sections, we see the many ways hegemony appears throughout our participants’ definition of CP. In this section, we revisit the three main hegemonic forces from our theoreti-

cal framework (rendering technical, constructionism, hacker culture and liberalism) and characterize how these three main fronts are reckoned with in CP.

There are many hegemonic forces at play in CS, such as colonialism, extractivism, racism, ableism, eugenics, and patriarchy, to name a few. We didn't have an intended theory in mind when analyzing our data, but after immersing ourselves in the data, we found that the hegemonic forces listed above (rendering technical/the God Trick, constructionism, and hacker culture/liberalism) provided the most parsimonious model of our data.

5.2.1 Resisting rendering technical

Of all the hegemonic forces in our interviews, our participants directly address rendering technical the most consistently. In both of the previous chapters, it's clear how our participants deliberately resist rendering technical and do so in a variety of ways. Some participants explicitly raise students' awareness of how their technology has an impact on society and "real people" (P6). Other participants, like P5 and 11, "widen the scope" of the content in the CS classes to include sociopolitical studies of CS to show how these aspects of CS are deeply connected. In yet another direction, some participants focus on supporting marginalized students in building context-specific software to address the social issues they face. All of these represent important ways our participants' resist rendering technical.

However, hegemony can be difficult to address in a comprehensive way. Indeed, our participants still, at times, adopt a rendering technical mindset. Consider P12, who warns how CP can be overly critical without making an effort to create social change. To address this, this participant believes students need to wield the tools of technology to counter harmful technology, like Facebook's unjust data collection. The issue at hand here is a social one: technology companies like Facebook, Amazon, Microsoft, or Google, benefit from under-regulation of social media and technology and therefore attain disproportionate resources and influence in society. The existence of these companies comes from a lack

of interventions society-wide to curb and parameterize the power of technology. However, P12's suggestion to write new programs that mitigate the harmful effects of these tech companies (like a browser plug-in to minimize Facebook's scraping of users' data) aligns with rendering technical as a superficial technological solution seems to be framed as a measure to address a larger, more complex social problem.

5.2.2 Tensions around constructionism

Our interviews reveal tensions around the adoption of constructionism in CP. Many participants, like P8, 12, and 13, find constructionism is an important and useful resource in CP. In particular, these participants narrow in on how CP and constructionism have a connected history and share many values and objectives. Indeed, these pedagogies are complimentary in their encouragement of getting students to do and create in society, as opposed to passivity. Furthermore, some participants describe constructionism as a useful tool to avoid weaknesses with CP, such as being overly rhetorical and lacking action.

However, other participants' conceptions of CP run counter to cooperation between constructionism and CP. Consider P9, who describes having students take ownership of the learning as a delicate process requiring careful hand-off between teacher and students. This is echoed by other participants, like P7, who ease students into taking autonomy of their learning. This is in sharp contrast to a constructionist way of learning, such as the P12 vignette, which frames constructionist learning as providing students with the tools to play with technology, which will result in them learning.

A potential danger in constructionist teaching of CS is it may only serve to reproduce patterns of privilege. Constructionism is built around the assumption of a "technically precocious young boy" [7]; and is set up to help these young auto-didacts excel, in turn reinforcing patterns of privilege in education. Consider P1's teaching. They mainly work with students of color in inner-city secondary and primary schools in the US. They describe how their students, to who they teach activism and CS, don't "primarily see themselves as technologists". This is a result of the pattern of exclusion and inclusion in CS. If

these students were left to their own devices to play and learn from technology, it's not immediately clear to us that they would see themselves as the right group to be using technology.

Not all students are socialized or ready to learn by breaking things [113]. Indeed, many groups of students, such as students of color, have been taught by society that breaking things is dangerous: consider the story of Ahmed Mohamed, a 14-year-old high school student in Irving, Texas, who reassembled a digital clock in his pencil case. Unfortunately, the teaching staff at his school believed he had constructed a homemade bomb and was subsequently arrested before authorities realized it was simply a clock [66]. And with that, constructionism fails to address racism by not reaching students in computing who are otherwise taught not to play with technology like white boys are.

Unstructured learning environments common in constructionist teaching may not be the best for all students or circumstances and pose an important contradiction with CP. Firstly, there are documented limitations of constructionism, including the transferability of skills learned through constructionism [97]. Although many educators and professors may resonate with constructionism, it is worth noting that professors and educators do not represent the average learner. Papert's work makes the assumption that a technically precocious young white boy is representative of learners writ large [7], without regard for the gendered, racialized, and classed dynamics involved in which students are given the opportunity to learn on their own; nor does it take into account identity formation, and how non-white men students often do not see themselves as technologists. Herein lies an important contradiction between constructionism and critical pedagogy: CP aims to address each student's experience of oppression as a consequence of their social identity, but constructionism assumes a theory of learning that universalizes the experience of the dominant student identity, i.e. technically precocious white boys. This contradiction poses a serious threat to the compatibility of constructionism with CP and any educators who wish to combine them need to think carefully and reflexively about how and where to combine them.

5.2.3 Resisting or co-operating with liberalism?

Finally, liberalism (as associated with hacker culture described in 1.4.4) is the last hegemony we often see in our data, with limited examples of it being addressed in CP. Indeed, many but not all of our participants' conceptions of CP, in particular their ideas of taking action in CP, fell in line with liberal ideals of individuality and even entrepreneurialism. On entrepreneurialism, one participant's approach to CP is creating and selling online professional development courses for teachers looking to integrate critical CS education in their teaching. This is something we were not expecting to see in our interviews given our own beliefs about CP. This lead us to reflect on the balance of pragmatism and dogmatism when doing CP, and points the tight rope our participants had to walk in doing CP will still conforming to the demands and expectations of living under capitalism.

Analyzing participants' visions of activism, we see more ways liberalism appears. Firstly, many participants in our study were very hesitant to engage in direct political action, including working with students to do activism. Consider P6, who "encourages" students to make a political change in their university but doesn't actively involve themselves in this process. Similarly, P9, who encourages students to take action in their context without actively organizing with students. Liberal values of individuality and opportunity can play a role here. From a liberal perspective, what matters is that those individual students have the opportunity to make a political change. Whether they choose to or not is their personal freedom. Furthermore, notions of individualism from liberalism can lead CP educators to feel like others' issues are not their problems to address. Similarly, individualism can over-emphasize the needs of the self and self-preservation, at the expense of the collective whole.

Hegemonic liberalism presents itself in pedagogies adjacent to CP too. Consider Yuen et al.'s articulation of CRP in CS [119]. The fourth tenant in their translation of CRP in CS calls for a pedagogy that enables students to "transcend any form of oppression" that students face. This is extremely individualistic without addressing the root causes of oppression, instead encouraging students to deal with it on their own.

P1 remains a notable figure in resisting liberal norms in CP. Their teaching was unique in how they actively encouraged and work with students to engage in direct action like protesting and sit-ins. This represents the kind of collective action necessary to address systemic issues in society by working collaboratively with students to face these systems instead of leaving it to students and hoping they do it.

Given students are much less privileged and have much less access to power almost guarantees that their success will be limited in addressing the systemic issues that lead to their oppression. Instead, if CP hopes to address liberalism, it must find a way to incorporate activism more meaningfully in the teaching of CS, paying particular attention to how students and teachers can work together to form a powerful political resistance. Only then will CP be better positioned to result in meaningful change, as opposed to hoping students will do it all themselves. At the same time, some educators find themselves with precarious employment (e.x. non-tenured professors). Educators looking to implement CP must also balance the security of their employment when attempting to do CP, and educators with higher employment security (e.x. tenured professionals) have a role to play in supporting educators with less security in their efforts to do CP.

5.3 CP and related pedagogies

5.3.1 CP and related social justice pedagogies

While we recruited participants who said they were *influenced* by critical pedagogy, many of our participants described themselves as being more in line with related pedagogies like culturally relevant pedagogy (CRP) and ethno-computing. These pedagogies have a good deal of overlap, and similarly seem to resist the adoption of a rendering technical framework in students. CRP, for instance, aims to support student growth by aligning teaching with each student's cultural identity [119]. Critical race theory, a related theory, looks to uncover and disrupt the systemic, hegemonic Whiteness of education, including in computer science [78]. Anarchist pedagogies, which vary greatly, still have a common

thread of resistance and skepticism towards state-run education programs [8]. Finally, student-centered pedagogy aims to disrupt the asymmetrical power relationship between students and teachers by centering students [116].

Many of the goals of these pedagogies have significant overlap with the goals of CP, with multiple participants in our study mentioning these pedagogies at least once. And in recent years, we have seen work published on anarchist CS education [62], culturally relevant pedagogy (CRP) [119], and ethno-computing pedagogies [45]. Indeed, we see a great deal of overlap between CRP and the pedagogy described by our participants. Take, for instance, plank 1, which calls for learning about students' backgrounds and incorporating them into the teaching process. P3's framing of being a "cultural anthropologist" is very much in line with CRP [65]. All in all, the above pedagogies have much in common.

5.4 Contributions

This thesis offers a number of contributions to the literature on both CP and critical CS education. Firstly, it provides concrete data on how to do CP in the CS context, addressing the lack of practice-focused research. Furthermore, although our participants were all from CS, the ontology that emerged from these interviews is likely useful for using CP in other domains. The ontology is important because it helps break down CP into smaller component parts. As I mentioned from my standpoint, a barrier for me doing CP was wondering where to start. Breaking CP down into parts facilitates the uptake of CP as practitioners looking to do CP can start from any point in the ontology and build their critical teaching in a way that works best for them. A final contribution is identifying the hegemonic forces that teachers face when trying to enact a more equitable CS education. Indeed, the hegemonic forces laid out in this thesis triangulate the forces that teachers must address in their effort to address racism, patriarchy, extractivism, ableism, and so on. Just as Dr. Patitsas and I note in our duo-autoethnography [80], rendering technical was the barrier that prevented us from addressing extractivism and environmental injustice

in our teaching. So too, can this thesis help other practitioners looking to address systems of inequality by narrowing in on the hegemonic forces they will face in this work.

5.5 Implications for CS educators[1]

Critical pedagogy offers a number of benefits for students, educators, and society. Students benefit from pedagogy where they have their voice heard: as one participant put it in our interviews: "[CP] means giving space for what you [i.e. students] enjoy and who you are, where you're coming from, into the projects you do and the stuff you learn" (P10). Centering students' motivations, especially how computing can benefit a students' community, is shown to improve students' motivation and sense of belonging, especially for those from marginalized backgrounds [43,69,90,110].

Indeed, critical pedagogy's benefits are even stronger for students who are already likely to feel disempowered by a computing education. For women, non-binary students, racial minorities, disabled students, and other minoritized groups in computing, it can be especially powerful to be invited take an active role in one's education. In other fields where CP has been deployed, it has improved retention of minoritized groups [82]. For example, developing a critical consciousness about gender issues in physics has improved how many women persist in physics [60].

Many educators can, understandably, feel nervous about getting political in the classroom. Our participants were no exception. For Participant 7: "[I was nervous] But it turns out that [students] really wanted to have this conversation."

Teaching in formal contexts, especially large classes, can feel alienating and dehumanizing for both students and educators. Participants noted that bringing more CP into their teaching empowered them to be more deliberate about their teaching. As on participant put it: "[CP helped me] see what role I was playing in my students' learning ... [CP]

[1]Large parts of this section borrow from *Critical Pedagogy in Practice in the Computing Classroom*, a forthcoming paper written by Dr. Patitsas and me

allowed me to have a discourse with myself and others about the decisions that I was making as a teacher."

Large classes present a specific challenge for those interested in CP. The vignettes we chose are all from small classroom settings, with some participants describing CP can be more difficult to do in a large class. Nevertheless, educators can do many CP-inspired things in large classes. Some examples from our participants were:

- Regularly "giving students a choice" (P7) or, say, in terms of assignments, course direction. Some participants structured their classes around final projects, where students are encouraged to work on a project that is meaningful to them.
- Incorporating readings about the social impacts of computing, paired with guided discussion prompts about the impacts of technologies.
- Talking explicitly about computing as a social institution, entangled with other institutions like racism, sexism, and white supremacy.
- Talking explicitly that CS education is political. For instance, one participant's first lecture in their CS class is titled "Your [CS] education is never neutral", or another participant exposes students to the capitalist motivations of CS education.
- Providing historical context for technologies presented in class - why are things the way they are, and what social processes shape our technologies? [5]
- Training TAs to carry out CP in labs. One participant trains TAs to bring criticality into their labs by role-modeling behaviour and questioning consistent with CP.

Several participants described their journey toward CP-inspired teaching as a "continually iterative" (P1) process that is ongoing. Participants began with one or two pieces - such as contextualized computing assignments - and iterated from there. They also described their pedagogy as varying from class to class - such as whether a class is large or small, the students' first-years versus fourth-years, and so on.

5.6 Limitations and future work

Limitations of our study come from the partiality of the research. Indeed, no one is outside the text, including the researchers who analyzed the interviews to identify hegemony. Having studied CS ourselves and taken courses that render technical, it is, without doubt, our analysis is limited itself by the ways hegemony continues to affect us as researchers. It's important that further work done by researchers identifying hegemony in CS remain reflexive and work with others to catch each other's blind spots. Another limitation discussed in Chapter 4, is how our participants all come from a CS background. From our interviews, we see an emergent ontology of CS that arises; however, this perceptive is partial in that it is only based on CS educators doing CP. Indeed, there are hegemonic forces unique to CS compared to other domains. By talking to practitioners of CP in different domains, a new ontology is likely to emerge.

In terms of work that remains to be done on the data collected for this study, more analysis is needed in exploring the difficulties participants faced in doing CP. Our interviews were rife with participants describing the way they were limited or blocked in their ability to do CP. Some examples include pushback from administration and students, lack of resources or guidance doing CP, or issues of ideological purity when to doing CP, just to name a few. Identifying the blockers that participants face doing CP can help draw attention to which changes need to be made in our institutions to make doing CP easier for teachers.

Further research in CP in CS should include observation and student perspectives. For instance, a future study could investigate CP using longitudinal observations in the classroom. Observing participants will help better articulate what CP looks like in the classroom and eliminate the say-do gap this work is susceptible to. Another novel direction subsequent research could investigate is how students perceive CP, such as what emotions come up while learning through CP. As our participants noted, CP can be a very emotional way of learning, so much so that one participant began to cry when thinking

about how oppressive systems of education can make students feel worthless and unintelligent. Exploring students' feelings can help educators looking to do CP better support students in this learning process, and potentially reveal more insights about hegemony and power.

Chapter 6

Conclusion

In this thesis, we explore the issues of justice and inequality execrated by technology and enabled by CS education. The emergence of unintended consequences as a result of technology is a growing and worrying trend [37]. Using theoretical concepts like Samantha Breslin's rendering technical, Haraway's "God trick", and liberalism ideology in CS, we expose how CS education manufactures a CS personhood that disregards technology's social and political consequences.

We explore relevant pedagogies, particularly Critical Pedagogy (CP), in an attempt to resist the apolitical worldview CS education enables. Ideally, a pedagogy that centers sociopolitical awareness and action in CS provides a worthwhile way to mitigate the creation of harmful tech. Instead of proposing yet another prescriptive and theoretical definition of CP in CS, one of the contributions from this thesis is a descriptive approach to critical CS education, synthesized from interviews with 13 self-describing pedagogues who are (at least) inspired by CP. These interviews seek to understand how our participants go about enacting CP in the classroom and what they consider to be important facets of this pedagogy.

Qualitative analysis of our interviews yields two important levels of analysis when conceptualizing CP. The first, described in Chapter 3, provides a rich description of CS by drawing on select vignettes from our interviews. This provides a very high detail

assessment of CP in CS. The second scope is wider, describing writ large the facets of CP described by our participants. From this analysis, we see 4 main facets of CP described in Chapter 4: 1) Engaging students on their terms, 2) changing the student-teacher dynamic, 3) centering power in your teaching, and 4) taking action.

The discussion section of our results describes how the concepts from our theoretical framing, like rendering technical and liberalism, are at times resisted and at other times supported in our participants' definition of CP. For instance, our participants actively work to resist hegemonic norms like rendering social issues into simple technical problems with simple technical solutions. However, some other forms of hegemony, like liberal conceptions of individualism, remain notably at play in our participants' notion of CP.

This research provides an important first step in the study of critical pedagogies in CS education and beyond. From this research, a number of new lines of inquiry arise. For instance, observation of critical pedagogues in the classroom can provide more accuracy regarding how the practices are employed in teaching. Furthermore, studies that center students' perspectives and emotions will shed light on how to best support students in the CP learning process. Ideally, with this triangulation, the CS education research community can continue to articulate a pedagogy that helps students and teachers resist apolitical tech and transform CS into a tool for environmental and social justice.

Bibliography

[1] *ACM Trans. Comput. Educ. 21*, 4 (2021).

[2] *ACM Trans. Comput. Educ. 22*, 3 (2022).

[3] ABRAHAMS, F. The application of critical pedagogy to music teaching and learning: A literature review. *Update: Applications of Research in Music Education 23*, 2 (2005), 12–22.

[4] ACKERMANN, E. Piaget's constructivism, papert's constructionism: What's the difference. *Future of learning group publication 5*, 3 (2001), 438.

[5] AKERA, A., ASPRAY, W., ET AL. *Using history to teach computer science and related disciplines*. Computing Research Association, 2004.

[6] ALEXANDER, R. *Culture and pedagogy*. Oxford, UK, 2000.

[7] AMES, M. G. Hackers, computers, and cooperation: A critical history of logo and constructionist learning. *Proceedings of the ACM on Human-Computer Interaction 2*, CSCW (2018), 1–19.

[8] ANTLIFF, A. *Anarchist pedagogies: Collective actions, theories, and critical reflections on education*. PM Press, 2012.

[9] APPLE, M. W. Can critical pedagogies interrupt rightist policies? *Educational theory 50*, 2 (2000), 229–254.

[10] APPLE, M. W. What policy can learn from the math wars. *Educational Policy 16*, 3 (2002), 448–456.

[11] APPLE, M. W. *Educating the right way: Markets, standards, God, and inequality*. Routledge, 2013.

[12] ASLAN TUTAK, F., BONDY, E., AND ADAMS, T. L. Critical pedagogy for critical mathematics education. *International Journal of Mathematical Education in Science and Technology 42*, 1 (2011), 65–74.

[13] BANKS, D. A., AND LACHNEY, M. Engineered violence: Confronting the neutrality problem and violence in engineering. *International Journal of Engineering, Social Justice, and Peace 5* (2017), 1–12.

[14] BARENDREGT, W., BECKER, C., CHEON, E., CLEMENT, A., REYNOLDS-CUÉLLAR, P., SCHULER, D., AND SUCHMAN, L. Defund big tech, refund community. *Tech Otherwise* (2021).

[15] BEHN, C., AND BAKKER, K. Rendering technical, rendering sacred: The politics of hydroelectric development on british columbia's saaghii naachii/peace river. *Global Environmental Politics 19*, 3 (2019), 98–119.

[16] BENJAMIN, R. *Race after technology: Abolitionist tools for the new jim code*. John Wiley & Sons, 2019.

[17] BENJAMIN, R. Race after technology: Abolitionist tools for the new jim code. *Social forces* (2019).

[18] BLIKSTEIN, P., AND WORSLEY, M. Children are not hackers: Building a culture of powerful ideas, deep learning, and equity in the maker movement. In *Makeology*. Routledge, 2016, pp. 64–79.

[19] BOLUDA, I. M., PATITSAS, E., AND MCMAHAN, P. What do computer scientists know about conflict minerals? In *Workshop on Computing within Limits* (2021).

[20] BORING, A., AND OTTOBONI, K. Student evaluations of teaching (mostly) do not measure teaching effectiveness. *ScienceOpen Research* (2016).

[21] BOWERS, C. A., ET AL. *Re-thinking Freire: Globalization and the environmental crisis*. Routledge, 2004.

[22] BRESLIN, S. *The making of computer scientists: rendering technical knowledge, gender, and entrepreneurialism in Singapore*. PhD thesis, Memorial University of Newfoundland, 2018.

[23] BREUNIG, M. Teaching for and about critical pedagogy in the post-secondary classroom. *Studies in Social Justice 3*, 2 (2009), 247–262.

[24] BRINKMAN, B., AND MILLER, K. W. The code of ethics quiz show. In *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education* (New York, NY, USA, 2017), SIGCSE '17, Association for Computing Machinery, p. 679–680.

[25] BROUSSARD, M. *Artificial unintelligence: How computers misunderstand the world*. mit Press, 2018.

[26] BRUCE, K. B., CUPPER, R. D., AND DRYSDALE, R. L. S. A history of the liberal arts computer science consortium and its model curricula. *ACM Transactions on Computing Education (TOCE) 10*, 1 (2010), 1–12.

[27] BURTON, E., GOLDSMITH, J., AND MATTEI, N. How to teach computer ethics through science fiction. *Commun. ACM 61*, 8 (jul 2018), 54–64.

[28] CALIFF, M. E., AND GOODWIN, M. Effective incorporation of ethics into courses that focus on programming. *SIGCSE Bull. 37*, 1 (feb 2005), 347–351.

[29] CANOSA, R. L., AND LUCAS, J. M. Mock trials and role-playing in computer ethics courses. In *Proceedings of the 39th SIGCSE Technical Symposium on Computer Science*

Education (New York, NY, USA, 2008), SIGCSE '08, Association for Computing Machinery, p. 148–152.

[30] CARLSON, D. Finding a voice, and losing our way? *Educational theory 48*, 4 (1998), 541.

[31] CHASE-DUNN, C., TAYLOR, P., ARRIGHI, G., COX, R., OVERBEEK, H., GILLS, B., FRANK, A. G., MODELSKI, G., AND WILKINSON, D. Hegemony and social change. *Mershon International Studies Review 38*, 2 (1994), 361–376.

[32] CHECKLAND, P. Soft systems methodology: a thirty year retrospective. *Systems research and behavioral science 17*, S1 (2000), S11–S58.

[33] CHRISTIANO, A., AND NEIMAND, A. Stop raising awareness already. *Stanford Social Innovation Review 15*, 2 (2017), 34–41.

[34] CLEAR, T. Critical enquiry in computer science education. *Computer Science Education Research: The Field and The Endeavour, Routledge Falmer, Taylor & Francis Group, London* (2004), 101–125.

[35] COCHRAN, G. L., HYATER-ADAMS, S., ALVARADO, C., PRESCOD-WEINSTEIN, C., AND DAANE, A. R. Social justice and physics education. In *Teaching and Learning for Social Justice and Equity in Higher Education: Content Areas*. Springer, 2021, pp. 125–147.

[36] COLEMAN, E. G., AND GOLUB, A. Hacker practice: Moral genres and the cultural articulation of liberalism. *Anthropological Theory 8*, 3 (2008), 255–277.

[37] DAO, D., W., H., WALLA, A.-A., VOCALFAN, RUBIEL, E., SCHREIBER, F., JAWORSKI, J., LIU, L., WILLIAMS, N., PAWLOWSKI, N., AMMANAMANCHI, P. S., STADLMANN, S., KÜHNE, S., TAIZ, DIETHE, T., JVMNCS, AND TWSL. daviddao/awful-ai: Awful ai - 2021 edition, Jan. 2022.

[38] DARDER, A. *A dissident voice: Essays on culture, pedagogy, and power*. Peter Lang New York, NY, 2011.

[39] DAVIS, J., AND WALKER, H. M. Incorporating social issues of computing in a small, liberal arts college: a case study. In *Proceedings of the 42nd ACM technical symposium on Computer science education* (2011), pp. 69–74.

[40] DAVIS, J., AND WALKER, H. M. Incorporating social issues of computing in a small, liberal arts college: A case study. In *Proceedings of the 42nd ACM Technical Symposium on Computer Science Education* (New York, NY, USA, 2011), SIGCSE '11, Association for Computing Machinery, p. 69–74.

[41] DIMITRIADIS, G., AND KAMBERELIS, G. *Theory for Education: Adapted from Theory for Religious Studies, by William E. Deal and Timothy K. Beal*. Routledge, 2006.

[42] DOORE, S. A., FIESLER, C., KIRKPATRICK, M. S., PECK, E., AND SAHAMI, M. Assignments that blend ethics and technology. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education* (2020), pp. 475–476.

[43] EDWARDS, D. Computational thinking and its mathematics origins through purposeful music mixing with african american high school students.

[44] EGLASH, R., BENNETT, A., O'DONNELL, C., JENNINGS, S., AND CINTORINO, M. Culturally situated design tools: Ethnocomputing from field site to classroom. *American anthropologist 108*, 2 (2006), 347–362.

[45] EGLASH, R., GILBERT, J. E., AND FOSTER, E. Toward culturally responsive computing education. *Communications of the ACM 56*, 7 (2013), 33–36.

[46] EUBANKS, V. *Automating inequality: How high-tech tools profile, police, and punish the poor*. St. Martin's Press, 2018.

[47] FERREIRA, R., AND VARDI, M. Y. Deep tech ethics: An approach to teaching social justice in computer science. In *Proceedings of the 52nd ACM Technical Symposium on*

Computer Science Education (New York, NY, USA, 2021), SIGCSE '21, Association for Computing Machinery, p. 1041–1047.

[48] FIESLER, C., GARRETT, N., AND BEARD, N. What do we teach when we teach tech ethics? a syllabi analysis. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education* (2020), pp. 289–295.

[49] FINCHER, S., AND PETRE, M. *Computer science education research*. CRC Press, 2004.

[50] FREIRE, P. Pedagogy of the oppressed. In *Toward a Sociology of Education*. Routledge, 2020, pp. 374–386.

[51] GARNER, R., AND HANCOCK, B. H. *Social theory: continuity and confrontation: A reader*, vol. 1. University of Toronto Press, 2014.

[52] GIBBS, N. E., AND TUCKER, A. B. A model curriculum for a liberal arts degree in computer science. *Communications of the ACM 29*, 3 (1986), 202–210.

[53] GOLDWEBER, M., DAVOLI, R., LITTLE, J. C., RIEDESEL, C., WALKER, H., CROSS, G., AND VON KONSKY, B. R. Enhancing the social issues components in our computing curriculum: computing for the social good. *ACM Inroads 2*, 1 (2011), 64–82.

[54] GOLDWEBER, M., IMPAGLIAZZO, J., BOGOIAVLENSKI, I. A., CLEAR, A., DAVIES, G., FLACK, H., MYERS, J. P., AND RASALA, R. Historical perspectives on the computing curriculum (report of the iticse'97 working group on historical perspectives in computing education). In *The supplemental proceedings of the conference on Integrating technology into computer science education: working group reports and supplemental proceedings* (1997), pp. 94–111.

[55] GROSZ, B. J., GRANT, D. G., VREDENBURGH, K., BEHRENDS, J., HU, L., SIMMONS, A., AND WALDO, J. Embedded ethics: integrating ethics across cs education. *Communications of the ACM 62*, 8 (2019), 54–61.

[56] GUR-ZE'EV, I. Toward a nonrepressive critical pedagogy. *Educational theory 48*, 4 (1998), 463.

[57] HALMAGHI, H. Learning computer science was hard. unlearning computer science is harder. Master's thesis, McGill University, 2019.

[58] HARAWAY, D. Situated knowledges: The science question in feminism and the privilege of partial perspective. *Feminist studies 14*, 3 (1988), 575–599.

[59] HARDING, S. *Whose science? Whose knowledge?* Cornell University Press, 2016.

[60] HAZARI, Z., POTVIN, G., LOCK, R. M., LUNG, F., SONNERT, G., AND SADLER, P. M. Factors that affect the physical science career interest of female students: Testing five common hypotheses. *Physical Review Special Topics-Physics Education Research 9*, 2 (2013), 020115.

[61] IMPAGLIAZZO, J., CAMPBELL-KELLY, M., DAVIES, G., AND LEE, J. A. N. History in the computing curriculum. *IEEE Annals of the History of Computing 21*, 1 (1999), 4–16.

[62] JONES, S. T., ET AL. We tell these stories to survive: Towards abolition in computer science education. *Canadian Journal of Science, Mathematics and Technology Education 21*, 2 (2021), 290–308.

[63] KAFAI, Y., PROCTOR, C., AND LUI, D. From theory bias to theory dialogue: embracing cognitive, situated, and critical framings of computational thinking in k-12 cs education. *ACM Inroads 11*, 1 (2020), 44–53.

[64] KO, A. J., OLESON, A., RYAN, N., REGISTER, Y., XIE, B., TARI, M., DAVIDSON, M., DRUGA, S., AND LOKSA, D. It is time for more critical cs education. *Communications of the ACM 63*, 11 (2020), 31–33.

[65] LADSON-BILLINGS, G. Toward a theory of culturally relevant pedagogy. *American educational research journal 32*, 3 (1995), 465–491.

[66] LAYCOCK, J. Who believed there was a bomb and when did they believe it? what ahmed mohamed's clock says about belief and moral panic. *Bulletin for the Study of Religion 44*, 4 (2015), 40–44.

[67] LEE, C. H., AND SOEP, E. None but ourselves can free our minds: Critical computational literacy as a pedagogy of resistance. *Equity & Excellence in Education 49*, 4 (2016), 480–492.

[68] LESSER, L. M., AND BLAKE, S. Mathematical power: Exploring critical pedagogy in mathematics and statistics. *Reinventing critical pedagogy: Widening the circle of anti-oppression education 159174* (2006).

[69] LEWIS, C., BRUNO, P., RAYGOZA, J., AND WANG, J. Alignment of goals and perceptions of computing predicts students' sense of belonging in computing. In *Proceedings of the 2019 ACM Conference on International Computing Education Research* (2019), pp. 11–19.

[70] LI, T. M. *The will to improve: Governmentality, development, and the practice of politics*. duke university Press, 2007.

[71] LIBOIRON, M. *Pollution is colonialism*. Duke University Press, 2021.

[72] LORD, S. M., PRZESTRZELSKI, B., AND REDDY, E. Teaching social responsibility: Conflict minerals module for a circuits class. In *2018 World Engineering Education Forum-Global Engineering Deans Council (WEEF-GEDC)* (2018), IEEE, pp. 1–6.

[73] LOVELL, H., AND MACKENZIE, D. Accounting for carbon: the role of accounting professional organisations in governing climate change. *Antipode 43*, 3 (2011), 704–730.

[74] MALAZITA, J. W., AND RESETAR, K. Infrastructures of abstraction: how computer science education produces anti-political subjects. *Digital Creativity* (2019), 1–13.

[75] MANN, S., LOPEZ, M., LOPEZ, D., AND SMITH, N. Educating for ICT4S: Unpacking sustainability and ethics of ICT student intakes. In *29th International Conference on Informatics for Environmental Protection and the 3rd International Conference ICT for Sustainability (EnviroInfo & ICT4S 2015)* (2015), Atlantis Press, pp. 229–241.

[76] MARTIN, C. D., HUFF, C., GOTTERBARN, D., AND MILLER, K. Implementing a tenth strand in the cs curriculum. *Commun. ACM 39*, 12 (dec 1996), 75–84.

[77] MARTIN, J. R. What should we do with a hidden curriculum when we find one? *Curriculum Inquiry 6*, 2 (1976), 135–151.

[78] MARX, S., AND PENNINGTON, J. Pedagogies of critical race theory: Experimentations with white preservice teachers. *International Journal of Qualitative Studies in Education 16*, 1 (2003), 91–110.

[79] MATTHEWS, C. Critical pedagogy in health education. *Health Education Journal 73*, 5 (2014), 600–609.

[80] MAYHEW, E. J., AND PATITSAS, E. Materiality matters in computing education: A duoethnography of two digital logic educators.

[81] MCARTHUR, J. Achieving social justice within and through higher education: The challenge for critical pedagogy. *Teaching in Higher Education 15*, 5 (2010), 493–504.

[82] MENCKE, P. D. *Responding to critical pedagogy: Marginalized students and the college classroom*. Washington State University, 2010.

[83] MERRIAM, S. B., AND TISDELL, E. J. *Qualitative research: A guide to design and implementation*. John Wiley & Sons, 2015.

[84] MISA, T. J. *Communities of Computing: Computer Science and society in the ACM*. Morgan & Claypool, 2016.

[85] MITCHAM, C. A historico-ethical perspective on engineering education: from use and convenience to policy engagement. *Engineering Studies 1*, 1 (2009), 35–53.

[86] MOORE, A. Teaching, learning and the curriculum: pedagogic and curricular alternatives. In *Teaching and Learning*. Routledge, 2012, pp. 152–185.

[87] MURPHY, P. Defining pedagogy. In *Equity in the classroom*. Routledge, 2003, pp. 17–30.

[88] NAFUS, D. 'patches don't have gender': What is not open in open source software. *New Media & Society 14*, 4 (2012), 669–683.

[89] NATHAN, L. P., KACZMAREK, M., CASTOR, M., CHENG, S., AND MANN, R. Good for whom? Unsettling research practice. In *Proceedings of the 8th International Conference on Communities and Technologies* (2017), pp. 290–297.

[90] NATIONAL ACADEMIES OF SCIENCES, ENGINEERING, AND MEDICINE AND OTHERS. *Cultivating interest and competencies in computing: Authentic experiences and design factors*. 2021.

[91] NIAS, J. Educational programming practices that inspire change: Social justice as situated in a computer programming course. In *2021 Conference on Research in Equitable and Sustained Participation in Engineering, Computing, and Technology (RESPECT)* (2021), pp. 1–5.

[92] NIELSEN, N. R. Social responsibility and computer education. In *Proceedings of the Second SIGCSE Technical Symposium on Education in Computer Science* (New York, NY, USA, 1972), SIGCSE '72, Association for Computing Machinery, p. 90–96.

[93] NÚÑEZ, A., MAYHEW, M., SHAHEEN, M., AND DAHL, L. Let's teach computer science majors to be good citizens. the whole world depends on it. *Edsurge* (2021).

[94] PAPERT, S. A. *Mindstorms: Children, computers, and powerful ideas*. Basic books, 2020.

[95] PARTHASARATHY, S., AND STILGOE, J. Episode 3: Considering ethical responsibility in science and technology ft. nicholas carr, 2019. The Received Wisdom.

[96] PARTHASARATHY, S., AND STILGOE, J. Equity in science and technology policy and the promise of vaccines ft.maya goldenberg, 2021.

[97] PEA, R. D. Logo programming and problem solving.

[98] PECK, E. The ethical engine: Integrating ethical design into intro computer science. *blog, Bucknell HCI 5* (2017).

[99] PUREWAL, T. S., BENNETT, C., AND MAIER, F. Embracing the social relevance: Computing, ethics and the community. In *Proceedings of the 38th SIGCSE Technical Symposium on Computer Science Education* (New York, NY, USA, 2007), SIGCSE '07, Association for Computing Machinery, p. 556–560.

[100] REGISTER, Y., AND KO, A. J. Learning machine learning with personal data helps stakeholders ground advocacy arguments in model mechanics. In *Proceedings of the 2020 ACM Conference on International Computing Education Research* (New York, NY, USA, 2020), ICER '20, Association for Computing Machinery, p. 67–78.

[101] REICH, R., SAHAMI, M., WEINSTEIN, J. M., AND COHEN, H. Teaching computer ethics: A deeply multidisciplinary approach. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education* (New York, NY, USA, 2020), SIGCSE '20, Association for Computing Machinery, p. 296–302.

[102] SALTZ, J., SKIRPAN, M., FIESLER, C., GORELICK, M., YEH, T., HECKMAN, R., DEWAR, N., AND BEARD, N. Integrating ethics within machine learning courses, 2019.

[103] SCHINDEL DIMICK, A. Exploring the potential and complexity of a critical pedagogy of place in urban science education. *Science Education 100*, 5 (2016), 814–836.

[104] SHIZHA, E. The interface of neoliberal globalization, science education and indigenous african knowledges in africa. *USA) Chairperson of the Foreign Language 1* (2010), 26.

[105] SIMON, R. I. *Teaching against the grain: Texts for a pedagogy of possibility*. Greenwood Publishing Group, 1992.

[106] SKIRPAN, M., BEARD, N., BHADURI, S., FIESLER, C., AND YEH, T. Ethics education in context: A case study of novel ethics activities for the cs classroom. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education* (2018), pp. 940–945.

[107] SKIRPAN, M., CAMERON, J., AND YEH, T. Quantified self: An interdisciplinary immersive theater project supporting a collaborative learning environment for cs ethics. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education* (2018), pp. 946–951.

[108] STINSON, D. W., BIDWELL, C. R., AND POWELL, G. C. Critical pedagogy and teaching mathematics for social justice. *The International Journal of Critical Pedagogy 4*, 1 (2012).

[109] SUCHMAN, L. Working relations of technology production and use. *Computer supported cooperative work 2*, 1-2 (1993), 21–39.

[110] THEOBALD, E. J., HILL, M. J., TRAN, E., AGRAWAL, S., ARROYO, E. N., BEHLING, S., CHAMBWE, N., CINTRÓN, D. L., COOPER, J. D., DUNSTER, G., ET AL. Active learning narrows achievement gaps for underrepresented students in undergraduate science, technology, engineering, and math. *Proceedings of the National Academy of Sciences 117*, 12 (2020), 6476–6483.

[111] VAKIL, S. Ethics, identity, and political vision: Toward a justice-centered approach to equity in computer science education. *Harvard Educational Review 88*, 1 (2018), 26–52.

[112] VAN HEERTUM, R. Marcuse, bloch and freire: Reinvigorating a pedagogy of hope. *Policy Futures in Education 4*, 1 (2006), 45–51.

[113] VOSSOUGHI, S., HOOPER, P. K., AND ESCUDÉ, M. Making through the lens of culture and power: Toward transformative visions for educational equity. *Harvard Educational Review 86*, 2 (2016), 206–232.

[114] WAHL, N. J. Yaatce—yet another approach to teaching computer ethics. *SIGCSE Bull. 31*, 1 (mar 1999), 22–26.

[115] WIKIPEDIA. Ontology (information science) — Wikipedia, the free encyclopedia. `http://en.wikipedia.org/w/index.php?title=Ontology%20(information%20science)&oldid=1118775608`, 2022. [Online; accessed 08-December-2022].

[116] WRIGHT, G. B. Student-centered learning in higher education. *International journal of teaching and learning in higher education 23*, 1 (2011), 92–97.

[117] WYKSTRA, S. Fixing tech's ethics problem starts in the classroom.

[118] YANG, K. W. *A Third University Is Possible*. University of Minnesota Press, 2017.

[119] YUEN, T., ARREGUIN-ANDERSON, M., CARMONA, G., AND GIBSON, M. A culturally relevant pedagogical approach to computer science education to increase participation of underrepresented populations. In *2016 International Conference on Learning and Teaching in Computing and Engineering (LaTICE)* (2016), IEEE, pp. 147–153.