COLLABORATIVELY LEARNING COMPUTATIONAL THINKING

BUSHRA T. CHOWDHURY

Dissertation submitted to the faculty of Virginia Polytechnic Institute and State University in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Engineering Education

Aditya Johri, Co-Chair
Vinod K. Lohani, Co-Chair
Dennis G. Kafura, Member
Elizabeth D. McNair, Member

6 JULY 2017
Blacksburg, VA

Keywords: Computational Thinking, Computational Concepts, Collaborative Learning, Social Interactions, Novice Learners
Collaboratively Learning Computational Thinking
Bushra T. Chowdhury

ABSTRACT

Skill sets such as understanding and applying computational concepts are essential prerequisites for success in the 21st century. One can learn computational concepts by taking a traditional course offered in a school or by self-guided learning through an online platform. Collaborative learning has emerged as an approach that researchers have found to be generally applicable and effective for teaching computational concepts. Rather than learning individually, collaboration can help reduce the anxiety level of learners, improve understanding and create a positive atmosphere to learning Computational Thinking (CT). There is, however, limited research focusing on how natural collaborative interactions among learners manifest during learning of computational concepts.

Structured as a manuscript style dissertation, this doctoral study investigates three different but related aspects of novice learners collaboratively learning CT. The first manuscript (qualitative study) provides an overall understanding of the contextual factors and characterizes collaborative aspects of learning in a CT face-to-face classroom at a large Southeastern University. The second manuscript (qualitative study) investigates the social interaction occurring between group members of the same classroom. And the third manuscript (quantitative study) focuses on the relationship between different social interactions initiated by users and learning of CT in an online learning platform Scratch™. In the two diverse settings, Chi’s (2009) Differentiated Overt Learning Activities (DOLA) has been used as a lens to better understand the significance of social interactions in terms of being active, constructive and interactive. Together, the findings of this dissertation study contribute to the limited body of CT research by providing insight on novice learner’s attitude towards learning CT, collaborative moments of learning CT, and the differences in relationship between social interactions and learning CT. The identification of collaborative attributes of CT is expected to help educators in designing learning activities that facilitate such interactions within group of learners and lookout for traits of such activities to assess CT in both classroom and online settings.
Collaboratively Learning Computational Thinking

Bushra T. Chowdhury

GENERAL AUDIENCE ABSTRACT

One of the overarching processes defining the future is the digital revolution, impinging on, reshaping, and transforming our personal and social lives. Computation is at the core of this change and is transforming how problems are defined, and solutions are found and implemented. Computer modeling, simulation and visualization software, Smart grid, and Software Defined Radio, are few examples where computation has allowed us to tackle problems from varied perspectives. Vast domains await discovery and mapping through creative processes of Computational Thinking (CT). CT is the thought process that enables us to effectively work in such a technology driven collaborative society. It provides us the ability to find the right technology for a problem and apply technology to resolve the problem.

Skill sets such as understanding and applying computational concepts are essential prerequisites for success in the 21st century. One can learn CT by taking a traditional course offered in a school or by self-guided learning through an online platform. This doctoral study investigates three different but related aspects of how new learners are learning CT. The first qualitative study provides an overall understanding of circumstantial factors that influence the learning in a CT face-to-face classroom at a large Southeastern University. The second qualitative study investigates how students in groups (in the same classroom setting) can help each other to learn CT. And the third quantitative study focuses on users’ learning of CT in an online learning platform Scratch™. Together, the findings of this dissertation study contribute to the limited body of CT research by providing insight on new learner’s attitude towards learning CT, collaborative moments of learning CT, and the differences in the relationship between social interactions and learning CT. The identification of collaborative attributes of CT is expected to help educators in designing learning activities that facilitate such interactions within a group of learners and look out for traits of such activities to assess CT in both classroom and online settings.
DEDICATION

To my family
ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my committee members Dr. Vinod Lohani, Dr. Aditya Johri, Dr. Dannis Kafura and Dr. Lisa McNair. During the course of last five years, I have benefited tremendously from your guidance and support. Thank you.

My gratitude extends to Dr. Holly Matusovich for all the inspiring and thoughtful discussions and suggestions. Thank you Holly, I could not have done it without you.

I would like to thank my family, for all the sacrifices and difficulties they had to go through. Thank you Roddur, Fahmi, Mridula and Ahmed.

Finally I would like to thank my parents. Whatever I am today, it is because of their support and guidance. Thank you Abba and Amma.
# TABLE OF CONTENTS

**CHAPTER 1 : COLLABORATIVELY LEARNING COMPUTATIONAL THINKING** ...... 1

1.1 Introduction ........................................................................................................... 1
1.2 Key Terms ............................................................................................................. 4
1.3 Statement of Purpose ............................................................................................ 5
1.4 Significance of the research .................................................................................. 7
1.5 Research Questions and Research Design ............................................................ 8
1.6 Scope of study ....................................................................................................... 11
1.7 Limitations of the study ....................................................................................... 11
1.8 Researcher bias ................................................................................................... 12
1.9 Summary ............................................................................................................... 12

**CHAPTER 2 : LITERATURE REVIEW** .................................................................... 13

2.1 Computational Thinking ...................................................................................... 13
2.2 Learning CT Online ............................................................................................ 18
2.3 Face-to-Face CT Learning Opportunities ............................................................. 19
2.4 Assessment of CT ............................................................................................... 20
2.5 Collaboratively Learning CT .............................................................................. 21
2.6 Summary ............................................................................................................. 23

**CHAPTER 3 : UNDERGRADUATE STUDENTS’ EXPERIENCES IN LEARNING CT**
FROM AN INTERDISCIPLINARY GENERAL EDUCATION COURSE ........................... 24

3.1 Abstract ............................................................................................................... 24
3.2 Introduction .......................................................................................................... 24
3.3 Background and Theoretical Framework ............................................................. 25
3.4 Research Design .................................................................................................. 27
3.5 Findings ............................................................................................................... 30
6.4 Research Contributions ................................................................. 136

CHAPTER 7 : COMPLETE REFERENCE ......................................................... 139

APPENDIX A: CT COURSES OFFERED BY DIFFERENT UNIVERSITIES ............... 147

APPENDIX B: INFORMAL LEARNING CALCULATIONS BY YANG ET AL. (2015) .... 149

APPENDIX C: WORKED EXAMPLE OF YANG’S (2015) MODEL .......................... 151

APPENDIX D: SCRATCH DATA TABLE NAMES AND DESCRIPTIONS ...................... 157
# LIST OF TABLES

Table 1.1 Research questions, data collection, analysis and outcome of dissertation study .......... 8
Table 2.1 CT practices (College Board & NSF, 2011, p 6-7) .................................................. 15
Table 3.1 Profile of students interviewed............................................................................. 29
Table 3.2 A priori code, informed by major influences of collaborative learning by Stahl (2010, p.6) ......................................................................................................................... 30
Table 4.1 Different activities that constitute as active, constructive or interactive and cognitive process associated with it.................................................................................................. 51
Table 4.2 Different types of roles of the observer (Creswell, 2008) ....................................... 54
Table 4.3 Participant’s cohort, gender, and major in the CT course .................................... 56
Table 4.4 Type and concept covered in classwork problems during observations............. 60
Table 4.5 Operationalized overt behaviors in the context of learning CT based on Chi (2009) .. 63
Table 4.6 Summary of overt activity episodes of students coded as Active, Constructive, Interactive and Struggling across cohorts ........................................................................................................ 80
Table 4.7 Average cohort interactions, classwork problems solved and grade received .......... 93
Table 5.1 Chi’s DOLA framework and cognitive process associated with each type of behavior ........................................................................................................................................ 104
Table 5.2 Operationalized definitions of active interactions in Scratch ............................... 106
Table 5.3 Operationalized definitions of constructive interactions in Scratch ...................... 106
Table 5.4 Sub categories of CT learning and corresponding blocks used to evaluate .......... 108
Table 5.5 Cluster wise information regarding number of projects, comments etc., that was further analyzed in this study ............................................................................................................. 110
Table 5.6 Summary statistics for measures of users in the Gaming cluster. Columns are included for the median (M), mean (\(\bar{x}\)), standard deviation (\(\sigma\)) and Range ......................................................... 117
Table 5.7 Summary Pearson correlation analysis of social interaction factors and CT learning of the Gaming cluster (columns include values of Pearson correlation coefficient) ................. 119
Table 6.1 Research questions, data, and summary of findings ............................................ 124
LIST OF FIGURES

Figure 1.1 Outline of the dissertation study ................................................................. 10
Figure 2.1 Environments where novice learners can learn CT ........................................ 17
Figure 3.1 Major influences of collaborative learning (Stahl, 2010, p.6) ............................ 26
Figure 3.2 Factors influencing non-computer science majors experiences in learning CT within small groups ............................................................................................................. 44
Figure 4.1 Different types of overt learning activities in the DOLA framework (Chi, 2009) ...... 50
Figure 4.2 Students of the CT class solving problems in a cohort during class ................. 55
Figure 4.3 Sample Blockly code ...................................................................................... 57
Figure 4.4 Python code in Blockly environment ............................................................... 58
Figure 4.5 Spyder programming environment .................................................................... 58
Figure 4.6 Adapted framework for categorizing naturally occurring CT overt behaviors ...... 68
Figure 4.7 Percentage count of different overt behaviors exhibited by the Alpha cohort ....... 69
Figure 4.8 Episode count of different overt behaviors exhibited by each student of the Alpha cohort .................................................................................................................................. 70
Figure 4.9 Percentage count of different types of active, constructive and struggling behaviors exhibited by each student of the Alpha cohort ............................................................................................................. 71
Figure 4.10 Percentage count of different overt behaviors exhibited by the Beta cohort ...... 72
Figure 4.11 Episode count of different overt behaviors exhibited by each student of the Beta cohort .................................................................................................................................. 73
Figure 4.12 Percentage count of different types of active, constructive and struggling behaviors exhibited by each student of the Beta cohort ............................................................................................................. 74
Figure 4.13 Percentage count of different overt behaviors exhibited by the Gamma cohort ...... 75
Figure 4.14 Episode count of different overt behaviors exhibited by each student of the Gamma cohort .................................................................................................................................. 76
Figure 4.15 Percentage count of different types of active, constructive and struggling behaviors exhibited by each student of the Gamma cohort ............................................................................................................. 77
Figure 4.16 Number of episodes of overt behavior exhibited by each cohort during observation .................................................................................................................................. 78
Figure 4.17 Exhibited overt behavior by different disciplinary students ......................... 79
CHAPTER 1: COLLABORATIVELY LEARNING COMPUTATIONAL THINKING

1.1 Introduction

According to the President George W. Bush’s Information Technology Advisory Committee (PITAC) report, computation is recognized as the third pillar of science, along with theory and experimentation (Benioff & Lazowska, 2005). Computation is transforming how problems are defined, and solutions are found and implemented. Computer modeling, simulation and visualization software, Smart grid, and Software Defined Radio, are a few examples where computation has allowed us to tackle problems from varied perspectives. Although computation is becoming increasingly critical across a range of domains, there is still little consensus on how to capture the different elements that comprise computation. In recent years, a conceptual framework has emerged termed as “Computational Thinking” (CT) incorporating the essential elements emphasizing the common and fundamental underlying process. Wing (2006, 2011) describes it as a thought process which allows one to understand a problem, design a solution and analyze results by applying concepts and methods commonly used by computer scientists.

In order to be successful in the 21st century, skill such as CT-- a starting point in the thought process, is emerging as indispensable not only in science and engineering but also across all disciplines (Bundy, 2007; Voogt, Erstad, Dede, & Mishra, 2013; Wing, 2011). In a minimalist sense, someone with the ability to use computation effectively will have an edge over someone who does not understand underlying computational methods and tools (ACM, 2014). Apart from being an essential professional competency, CT also provides personal empowerment (National Research Council, 2010). It facilitates smooth integration to societal changes based on technological transformation. The understanding of computational concepts helps better comprehend and assess common issues such as digital voting, digital privacy and rights, online payment systems etc. This in turn helps in advocating and preserving one’s own rights and privileges and questioning/challenging service providers and regulators. Being able to apply computational methods and tools to solve daily problems also contributes to personal satisfaction and empowerment. Thus, CT is increasingly seen as a fundamental skill similar to reading, writing and arithmetic that everyone can benefit from (Wing, 2006).
This new framework is driving public and private institutions and regulatory authorities to incorporate CT within policy documents, strategic plans and curriculum (Clough, 2004; Virginia Tech, 2012; Wing, 2011). In order to provide every student with the opportunity to learn computer science, former President Obama announced his Computer Science for all initiative (CSforALL) in January 2016 proposing over 4 billion dollars to fund states and districts to support and train CS teachers. At the university level, various CT initiatives are also being put in place. For example, Virginia Tech has incorporated CT as a competency that all of its graduates should acquire.

“Virginia Tech is committed to a progressive agenda that provides the educational opportunities, computational infrastructure, and learning spaces necessary to prepare students and faculty to excel in this environment. Emphasis will be given to developing core competencies in computational thinking, information literacy, and analytical Methods” (Strategic plan of Virginia Tech, 2012, p.5)

Both formal and informal CT learning opportunities are being introduced at the K-12 and university levels (Steve Cooper, Grover, Guzdial, & Simon, 2014; Grover & Pea, 2013). One can learn computational concepts by taking a traditional course offered in a school or by self-guided learning through an online platform. At the K-12, CT is being integrated within various disciplinary (e.g., Mathematics, Science, English, Music and Journalism) activities and also being offered as a separate course, e.g. course in AP CS (Grover & Pea, 2013; National Research Council, 2012). In higher education, some universities have been integrating CT modules throughout their undergraduate programs while others are offering it as a semester long course. Some of these courses are being offered with a particular discipline’s viewpoint, and others as a general education course designed for all non-computers science majors (Kafura, Bart, & Chowdhury, 2015; NRC, 2011). Whatever the form of learning, computational concepts can be difficult to fully comprehend for novices (N. C. Brown & Altadmri, 2014). For learners coming from a variety of backgrounds, CT needs to be introduced and taught in a manner that eases and facilitates the learning process. Creative game design, media development, and interacting with museum exhibits are few examples of approaches that researchers have used to lower the bar for
learners, although they vary in appeal and effectiveness (Brennan & Resnick, 2012; Chowdhury, Blanchard, Cameron, & Johri, 2015; Repenning, Webb, & Ioannidou, 2010).

Collaborative learning has emerged as a technique that researchers have found to be generally applicable and effective for teaching Computer Science concepts (Kavitha & Ahmed, 2015; McDowell, Werner, Bullock, & Fernald, 2002). Rather than learning individually, collaboration can help reduce the anxiety level of learners, improve understanding and thus create a positive atmosphere for learning CT (Hmelo-Silver, 2013). In a collaborative setting learners can engage in CT by developing representations, debugging processes, and so on, resulting in a collective process of discovery that is not only more welcoming for learners but also richer than trying to learn individually (National Research Council, 2010, p.27). This collaborative dimension of CT can be observed in social interactions among learners: for example, when two or more students code and debug a program together in a class or when students share, comment or reuse pre-existing code from each other in an online learning platform. Studies of online users in the CT learning platform Scratch have also revealed the influences between social interactions and learning of CT (Brennan & Resnick, 2012; Dasgupta, Hale, Monroy-Hernández, & Hill, 2016; Fields, Giang, & Kafai, 2013; Sylvan, 2010; Velasquez et al., 2014).

According to Miyake and Kirschner (2014), social interaction taking place in collaborative learning contributes to classroom knowledge construction. To understand collaborative process requires not only an understanding of how individuals perceive, remember, and solve problems, but also how they interact with their social and physical environments and participate in the collective process of problem solving and knowledge building (Jeong, 2013). From a situative perspective, the success of a learning experience relies on the social interaction and kinesthetic activity learners are involved in. However, the social part of learning is considered so natural that educators and researchers assume that they do not need to be addressed specifically. Also, though learning environments may be designed to foster social interactions, the mechanisms of social interactions and their contributions to learning are still not understood well.

The dissertation focuses on collaborative learning of CT as manifested in social interactions and investigates how such collaboration impacts learning of CT. This chapter
provides definitions of key terms, introduction to statement of purpose, an overview of research questions and analysis methodologies of the dissertation study.

1.2 Key Terms

The following section defines key terms used in this dissertation proposal.

- **Computational Thinking**
  “Computational thinking is the thought process involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent” (Cuny, Snyder, & Wing, 2010)

- **Computational concepts**
  Abstraction, iteration, conditional logic, algorithm, functions etc.

- **Collaborative learning**
  “Collaborative learning is a situation in which two or more people learn or attempt to learn something together” (Dillenbourg, 1999, p. 1)

- **Social interactions/ overt behavior/ exhibited behavior**
  Social interactions in collaborative learning have been characterized as describing, explaining, predicting, arguing, critiquing, evaluating, explicating and defining (Kreijns, Kirschner, & Jochems, 2003; Ohlsson, 1996). In this dissertation study, any external behavior of a learner that can be noticed by an observer was considered as social interaction/overt behavior/exhibited behavior.

- **Novice learner**
  Novices are defined as users who have had little or no previous experience with computers (Mayer, 1981, p. 123).

- **Scratch online community**
  Scratch is a free educational programming language that was developed by the Lifelong Kindergarten Group at the Massachusetts Institute of Technology (MIT). Scratch is designed to be fun, educational, and easy to learn. It has the tools for creating interactive stories, games, art, simulations, and more, using block-based programming. Users program in Scratch by dragging blocks from the block palette and attaching them to other blocks like a jigsaw puzzle (Lifelong Kindergarten, 2007).
1.3 Statement of Purpose

The purpose of this doctoral study is to better understand how the collaborative processes, specifically social interactions, influence the learning of basic computational concepts of novice learners. Collaborative CT is defined here as a group of learners engaged in learning basic computer science concepts (abstraction, iteration, conditional logic, and algorithm). Social interactions between group members are defined as externally observable behaviors i.e., physically doing something, verbally saying something, sharing something, etc. This study has three main objectives as listed below:

1. To better understand the collaborative aspects of learning CT in a face-to-face setting
2. To better understand the collaborative influence of social interactions and learning of CT in a face-to-face setting
3. To better understand the collaborative influence of social interactions and learning of CT in an online setting

The first objective of this study, resulting in the first manuscript (chapter 3), is to better understand the collaborative aspects of learning CT in an undergraduate classroom setting. A suitable site for investigating such a classroom is at Virginia Tech. The strategic plan of Virginia Tech (Virginia Tech, 2012) stressed the need for graduates to acquire core competencies in CT, information literacy, and analytical methods. Starting fall 2014, in line with the strategic plan of the university (stated in introduction of this chapter), Virginia Tech has been offering a general education course titled “Introduction to CT” to students from all disciplines (e.g., Political Science, English, Sociology, Animal Science, Physics, Engineering, and Business). During the fall 2014, an ethnographically informed pilot study was conducted to better understand how these undergraduates were learning CT from this course (Kafura et al., 2015). Findings from the study suggest that students found social interactions within groups to be useful in learning CT and that students’ disciplinary background influenced such learning (Kafura et al., 2015). Manuscript 1 of this dissertation documents the experiences novice non-CS majors had while learning CT in an undergraduate general education course. In order to gain an overall understanding of the contextual factors and characterize collaborative aspects of learning within
a CT classroom setting, Stahl (2013)’s framework of “major influences of collaborative learning” was used. Data used in this study were qualitative interview data of students enrolled in the CT class. The findings of this manuscript provide better understanding of how non-CS majors experience learning CT and what aspects of the collaborative learning the students found to be beneficial.

Manuscript 2 (chapter 4) of this dissertation aims to better understand the collective group process of learning CT in the classroom setting. Chi’s DOLA framework (2009) was used to analyze observation data of students interacting with each other in small groups in the same CT undergraduate class (discussed for manuscript 1 above). The DOLA framework allowed the researcher to categorize interactions and overt behaviors based on their suggestive cognitive value. The data used in this part of the study were qualitative, collected from video recordings of groups working during the classes. The outcomes, documented in Manuscript 2, operationalize the social interactions occurring between group members in a face-to-face class room setting. They also illustrate and categorize collaborative moments of learning CT. The findings suggest struggling, active and constructive social interactions that educators should keep in mind while designing collaborative behaviors that promote learning CT in a classroom setting.

Manuscript 3 (chapter 5) is focused on better understanding the collaborative aspects of learning CT in an online learning Community, Scratch, discussed earlier. Scratch (www.Scratch.mit.edu) is an online community and social networking forum. In this platform, one can code games, animations, and stories using a media-based programming language. In Scratch, one uses ‘blocks’, which are puzzle-piece shapes used to create code. The blocks connect to each other like a jigsaw puzzle, where each block represents a particular programming concept (e.g., if, do-if, repeat, end). A ‘project’ in Scratch can be a game or an animation created by a Scratch user by mixing different kinds of blocks. Currently the Scratch community has over 10 million projects shared and 7 million users. The Scratch community database from years 2007-2012 is available for analysis. By focusing on a previously identified cluster of Scratch users (Gelman, Beckley, Johri, Domeniconi, & Yang, 2016), the study investigates learning and interaction patterns and computational concepts (blocks) used by the members in each group. Similar to the classroom study (manuscript 2), this study also uses Chi’s DOLA framework to categorized social interactions initiated by online novice learners. Data used in this part of the study are mostly quantitative, generated from the online learning system. The findings of
Manuscript 3 illustrate the relationship between different social interactions initiated by users and learning CT. The findings of this study suggest assessment indicators that could be used to evaluate users’ online interaction patterns and learning of CT concepts.

1.4 Significance of the research

This study examines CT in a collaborative context whereas prior work has mostly focused exclusively on the CT skill one had learned (Brennan & Resnick, 2012; Dasgupta et al., 2016; Matias, Dasgupta, & Hill, 2016; Moreno-León, Robles, & Román-González, 2015). Theories of situated learning and distributed cognition view learning as a function of social context and interaction (Greeno, Collins, & Resnick, 1996). Thus, in order to better understand learning of CT from a situative perspective, it is vital to examine it in a collaborative context and focus on the social interactions between group members. These interactions are not only restricted to face-to-face interactions but also those between learners in online platforms. Online interactions via an email or liking someone else’s project can also be considered social interactions (Kreijns et al., 2003). This study focuses on naturally occurring interactions exhibited by novice learners while learning CT in collaborative settings. By examining such social interactions between learners, this study intends to characterize social interactions and their influences on learning CT. In doing so, the outcome of the study will provide recommendations for educators with respect to assessing and designing learning activities that promote learning CT in collaborative settings. It also intends to provide guidelines for CT learners as well. These guidelines will provide suggestions regarding how to effectively engage in collaboratively learning CT. The differences between classroom learning of CT vs. online learning of CT would also provide educators with better understanding of the limitations and affordances of each setting.
1.5 Research Questions and Research Design

The purpose of the dissertation study is to better understand how the collaborative process of learning, specifically social interactions, can influence the learning of basic computational concepts. For this purpose, three research questions were designed. Table 1.1 provides the details.

Table 1.1 Research questions, data collection, analysis and outcome of dissertation study

*Overarching research question*- How do social interactions of novice CT learners influence the learning of basic computational concepts?

<table>
<thead>
<tr>
<th>Research Question</th>
<th>Data Collection</th>
<th>Analysis</th>
<th>Outcome</th>
</tr>
</thead>
<tbody>
<tr>
<td>RQ1: How do novice learners/non-CS majors describe their experiences of learning CT in a general education course?</td>
<td>Individual interviews (10 students majoring in different disciplines)</td>
<td>Qualitative analysis: Coding informed by Stahl’s major influences of collaborative learning (Stahl, 2010)</td>
<td>Manuscript 1: Understanding of novice learners’/non-CS majors’ perspectives of learning CT in a general education CT course</td>
</tr>
<tr>
<td>RQ2: How do novice CT learners naturally interact with group members while learning CT in a classroom setting?</td>
<td>Observations (video recordings of 3 groups working together during class time, a total of 180 minutes of video recording)</td>
<td>Qualitative analysis: Coding of social interactions informed by Differentiated Overt Learning Activities (DOLA) framework (Chi, 2009)</td>
<td>Manuscript 2: Understanding the collaborative aspects (e.g., social interactions in terms of active vs. constructive) of learning CT in a classroom setting</td>
</tr>
</tbody>
</table>
RQ3: Do social interactions exhibited by novice CT learners in an online learning platform (Scratch) lead to difference in learning of CT?

Dataset of the Scratch learning platform (users identified by Gelman et al. (2016) a total of 4500 users’ data was used for analysis)

Quantitative analysis: Classification of social interactions informed by Differentiated Overt Learning Activities (DOLA) framework (Chi, 2009)

Learning CT was calculated using Yang et al. (2015) model of informal learning

Correlational analysis

Manuscript 3: Understanding the collaborative aspects (e.g., social interactions in terms of active vs. constructive) of learning CT in an online open platform

In order to better understand the significance of social interactions exhibited by novice CT learners, this study investigated two diverse settings (undergraduate face-to-face class and open online platform). The first two research questions were related to the face-to-face class room setting and the third research question was concerned with the online learning environment. Figure 1.1 illustrates the outline of this dissertation study.
A qualitative study investigating the first research question (RQ1) focusing on ‘novice CT learners’ experiences of learning CT in a classroom setting’ has been described in chapter 3. In order to answer the second research question (RQ2), ‘characterizing social interactions that lead to collaborative moments of learning CT in a classroom setting’, a qualitative ethnographically informed research study was conducted. The findings addressing RQ2 have been presented in chapter 4. The third research question (RQ3), ‘Do social interactions exhibited by novice CT learners in an online learning platform lead to difference in learning of CT’, has been addressed in a quantitative study conducted that focused on previously identified users (Gelman et al., 2016) in the online learning community, Scratch. The findings of this study have been described in chapter 5. In the context of this dissertation, using ‘chapter’ is more appropriate than manuscript.
1.6 Scope of study

The purpose of the dissertation study is to better understand how the collaborative process of learning, specifically social interactions, influence the learning of basic computational concepts. The study neither intends to measure collaborative learning as an outcome, nor measure the influence of collaboration learning on individual learning. Rather, the study explored the role of naturally occurring social interactions in learning CT in collaborative settings. The situative perspective views learning “as distributed among people and their environments, including objects, artifacts, tools, books, and the communities which they are part of” (Greeno et al., 1996, p. 20). From this perspective, knowledge is not something an individual possesses or stores in the brain but is present in all that they do (Johri & Olds, 2011, p. 155). A learner develops conceptual knowledge by doing tasks that a community values and adheres as work, and this is manifested in the way a learner interacts with others in a community. Thus, in order to better understand learning of CT from a situative perspective, learners must be observed in a setting where they are working towards understanding and applying concepts of CT, focusing on social interactions between group members.

Although CT is considered important for learners of all ages and disciplines, this dissertation is centered on novice learners and those having limited or no prior computer programming knowledge. This is because the objectives of the study were to provide recommendations on how to better teach and create a more effective learning environment for learners, specifically those who neither have the background nor are traditionally accustomed to learning concepts of computer science.

1.7 Limitations of the study

The limitations of this study are that the participants and the course contents of the two investigating settings are not the same. Furthermore, there are very limited self-reported information (e.g., age, sex) of the participants of the online learning community. However, the study did not intend to draw any generalized conclusions comparing the two settings. Rather it focused on factors influencing collaborative learning of CT, specifically social interactions in two diverse settings and provided recommendations for each setting separately.
1.8 Researcher bias

The author of this dissertation study is a graduate student with an undergraduate degree in computer science and experience in teaching Computer Science courses. In order to limit the researcher’s bias, various reliability and trustworthiness measures (Creswell, 2008; Patton, 2001) were taken, which have been further discussed in the methods section of each chapter.

1.9 Summary

The purpose of this dissertation study is to better understand the collaborative process of learning CT, specifically focusing on the naturally occurring interactions (social interactions) between group members. This study captures non-CS majors’ experiences of learning CT in a general education course. In doing so the study also explored factors that influence learning CT in collaborative settings. Student’s disciplinary background, prior experience in learning CT, willingness to help others and the atmosphere of the learning community influenced learning of CT. The study also categorizes social interactions exhibited by novice CT learners while collaboratively learning CT as struggling/active/constructive in the face-to-face setting and active/constructive in the online learning platform. These are important contributions to computer education and CT literature since there is limited study on the collaborative process of learning CT illustrating the role of social interactions.

The rest of this dissertation is structured as follows. Chapter 2 gives introduction to relevant terms in the context of this study such as Computational Thinking, collaborative learning, etc. Chapter 3, 4 and 5 describe research design, data collection and analysis procedures, measures taken for ensuring reliability and trustworthiness and findings of research questions 1, 2 and 3, respectively. Chapter 6 summarizes the findings and contributions of all three research questions.
CHAPTER 2: LITERATURE REVIEW

This dissertation study investigates learning of CT in collaborative settings: online and in a classroom. The focus of this study is on novice learners, those with no prior experience in programming, and the social interactions they exhibit in the learning process. This chapter presents a review of literature on CT, settings that promote novices to learn CT and features of social interactions in the context of collaborative learning.

2.1 Computational Thinking

The term Computational Thinking (CT) has gained momentum since the seminal Viewpoint article written by computer scientist Jeannette Wing in 2006. According to Wing:

“Computational thinking involves solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science” (Wing 2006, p. 33)

In her article, Wing defined “computational thinking” as “a universally applicable attitude and skill set everyone, not just computer scientists, would be eager to learn and use” (p. 33). The notion of CT is not new; it has been inspired by Seymour Papert and Alan Perlis. Papert’s notion of procedural thinking and research on LOGO programming for children in the 1980’s (Papert, 1980) and Perlis’s recommendation for college students of all disciplines to learn programming and the “theory of computation” in the 1960s laid the foundation for computational thinking (Guzdial, 2008).

Since Wing’s influential article in 2006, the term CT “has caught the attention of a broad academic community” (Grover & Pea, 2013, p. 38). In 2010, Cuny, Snyder, & Wing re-defined CT as:

“Computational thinking is the thought process involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent” (Cuny et al., 2010)

Aho defined CT “to be the thought processes involved in formulating problems so their solutions can be represented as computational steps and algorithms” (2012, p. 832). The National Research Council’s report on the ‘Scope and Nature of Computational Thinking’ refers to CT as
being a range of concepts, applications, tools, and skill sets; a language, a process of manipulating abstraction, and also a cognitive tool. The notion of data abstraction, algorithmic thinking, debugging, recursion, problem decomposition, heuristic reasoning, and reduction have been considered as domains of CT (National Research Council, 2010; Weinberg, 2013). The various definitions of/approaches to CT have been aggregated in terms of competencies and operationalized by experts from different disciplines and professions (Grover & Pea, 2013; NRC, 2010; Weinberg, 2013).

International Society for Technology Education & Computer Science Teachers Association gives the following description:

“Computational thinking is a problem-solving process that includes (but is not limited to) the following characteristics:

1. Formulating problems in a way that enables us to use a computer and other tools to help solve them.
2. Logically organizing and analyzing data
3. Representing data through abstractions such as models and simulations
4. Automating solutions through algorithmic thinking (a series of ordered steps)
5. Identifying, analyzing, and implementing possible solutions with the goal of achieving the most efficient and effective combination of steps and resources
6. Generalizing and transferring this problem solving process to a wide variety of problems”

The College Board and National Science Foundation (http://www.csprinciples.org/) has formulated the definition of CT into 6 main categories of ‘CT practices’ as:

1. Connecting computing
2. Developing computational artifact
3. Abstraction
4. Analyzing problems and artifacts
5. Communicating
6. Working effectively in teams
Table 2.1 explains the 6 main categories defined by College Board and NSF and student outcomes corresponding to each category

Table 2.1 CT practices (College Board & NSF, 2011, p 6-7)

<table>
<thead>
<tr>
<th>Main category</th>
<th>Students are expected to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connecting computing</td>
<td>1. Identify impacts of computing</td>
</tr>
<tr>
<td></td>
<td>2. Describe connections between people and computing</td>
</tr>
<tr>
<td></td>
<td>3. Explain connections between computing concepts</td>
</tr>
<tr>
<td>Developing computational artifact</td>
<td>1. Create an artifact with a practical, personal, or societal intent</td>
</tr>
<tr>
<td></td>
<td>2. Select appropriate techniques to develop a computational artifact</td>
</tr>
<tr>
<td></td>
<td>3. Use appropriate algorithmic and information-management principles</td>
</tr>
<tr>
<td>Dealing with/Understanding abstraction</td>
<td>1. Explain how data, information, or knowledge are represented for computational use</td>
</tr>
<tr>
<td></td>
<td>2. Explain how abstractions are used in computation or modeling; identify abstractions</td>
</tr>
<tr>
<td></td>
<td>3. Describe modeling in a computational context.</td>
</tr>
<tr>
<td>Analyzing problems and artifacts</td>
<td>1. Evaluate a proposed solution to a problem</td>
</tr>
<tr>
<td></td>
<td>2. Locate and correct errors</td>
</tr>
<tr>
<td></td>
<td>3. Explain how an artifact functions; and justify appropriateness and correctness</td>
</tr>
<tr>
<td>Communicating</td>
<td>1. Explain the meaning of a result in context</td>
</tr>
<tr>
<td></td>
<td>2. Describe computation with accurate and precise language, notation, or visualizations</td>
</tr>
<tr>
<td></td>
<td>3. Summarize the purpose of a computational artifact</td>
</tr>
</tbody>
</table>
As noted in Table 2.1, the operationalized versions of CT have been further broken down into a set of skills and competencies that students should be able to perform and to demonstrate the ability to think computationally.

The definitions of CT provided by Wing (2006, 2011, 2014) and the versions (College Board & NSF, 2011) discussed above, emphasize CT as an ability to formulate and solve problems using information representations and automated processing. Abstractions, iterations, conditional logic, algorithmic thinking, and parallel computing are considered fundamental concepts of CT. The cognitive process of understanding these concepts involve hierarchical and process oriented thinking. The literature also establishes that the concept of CT can be applied to a wide variety of problems in different disciplines (ACM, 2014; NRC, 2011; Wing, 2011).

That CT is a common underlying thinking process has been acknowledged by Wing (2014, 2011, 2006); Bundy (2007). According to Wing (2011) “if computational thinking will be used everywhere, then it will touch everyone directly or indirectly” (p.3720). Bundy (2007) phrased the universal applicability of CT as “Computational thinking is influencing research in nearly all disciplines, both the sciences and the humanities” (p. 67). According to CSTA and ISTE (2011), it is important for students to realize that CT is important for them, recognize and use computational concepts within their own discipline/interest, and realize the variety of problems and scenarios CT can be used in (CSTA & ISTE, 2011).

The overarching notion is that CT can be beneficial for learners of all ages and disciplines. From Engineering to English majors, from kids to adults, CT has been considered as an essential 21st century skill that everyone should possess (Voogt et al., 2013).

CT can be regarded both as an individual as well as a group phenomenon (National Research Council, 2010). Members in groups can “engage in CT to develop representations,
debug processes, and so on, resulting in a collective process of discovery that is richer than that of any single individual” (National Research Council, 2010, p.27). This collaborative dimension of CT can be observed in social interactions between group members: for example, when a group of students code and debug a program together in a class or when students share, comment or reuse pre-existing code from each other in an online learning platform.

Based on the literature reviewed above this study operationalizes CT as the ability to understand and apply computational concepts (e.g. abstraction, iteration, conditional logic, algorithms) to solve problems. Focusing on the collaborative perspective of learning of CT, this study investigates two diverse settings: learning CT online and in a college classroom. The study also focuses on novice CT learners, those who have had no prior programming experience.

The following sections on the literature review thus focus on online learning, the college setting and the collaborative aspects of learning CT. Figure 2.1 illustrates settings where novice learners can learn CT. Sections 2.2 and 2.3 further discuss research findings related to learning CT online and in classroom settings respectively. Section 2.4 discusses assessment strategies in CT. Section 2.5 provides an overview of research related to collaborative learning.

![Figure 2.1 Environments where novice learners can learn CT](image-url)
2.2 Learning CT Online

Most of the literature concerning learning CT online and learning to code for the first time are on graphical programming environments such as Scratch, Alice; from online interactive educational platforms such as Code Academy, Khan Academy; and from introductory computer science and programming courses offered on Massive open online courses (MOOCs) such as Coursera, Edx, etc.

2.2.1 GRAPHICAL PROGRAMMING ENVIRONMENTS

The graphical programming environments (e.g. Scratch, Alice, Blockly) are platforms with programming tools that allow users to create projects on their own or by collaborating with others (Sylvan, 2010). Although some guided tutorials are offered by the platforms, learning is mostly free form and depends on the user. Studies on graphical programming environments have mostly investigated the effectiveness of visual programming as a tool for introducing programming to novice learners (Brennan & Resnick, 2012; Stephen Cooper, Dann, & Pausch, 2000; Weintrop & Wilensky, 2015b). They have also examined how learners switch between visual and text-based representations of code, and how learning visual programming first influences learning text based programming later (Weintrop & Wilensky, 2015a). Data collected by these environments are allowing researchers to look at user interest (Agrawal, Jain, Kumar, & Yammiyavar, 2014; Gelman et al., 2016), CT learning patterns (Brennan & Resnick, 2012; Meerbaum-Salant, Armoni, & Ben-Ari, 2011), and various collaborative aspects that influence learning of CT(Brennan, Monroy-Hernández, & Resnick, 2010; Dasgupta et al., 2016; Yang et al., 2015).

2.2.2 INTERACTIVE EDUCATIONAL ENVIRONMENTS

Interactive educational environments (e.g. Khan Academy, Code Academy) are platforms where students or anyone interested in learning how to code can watch short videos, go through short interactive tutorials and self-assess one’s own learning by solving problems generated by the platform. A structured/guided path of learning is usually offered by these platforms. However, the learner can also pick and choose a starting point. Studies on online interactive educational platforms have investigated learning behavior and skill progression (Murphy, Gallagher, Krumm, Mislevy, & Hafter, 2014), assessment strategies (Morrison & DiSalvo, 2014;
Sohrabi & Iraj, 2016), user roles in online communities, help seeking behaviors and promptness and correctness of answers in community (Teo & Johri, 2014) forums etc.

2.2.3 MOOCs

MOOCs are providing learners (approximately 58 million in 2016) all over the globe access to college level courses online for free or for a minimal cost (Haber, 2014). In 2015, courses in computer science and related to programming increased by 10%. In recent years, science related courses offered on MOOCs have gained significant popularity. Courses on MOOCs are usually structured, video based, and shorter in length (2-4 weeks each) than traditional college level courses. There are some courses that are self-paced. Assessment of learning is usually automated or crowd sourced/peer reviewed. Studies on MOOCs have been investigating the high student dropout rates, effective automated assessment strategies, cultural differences and motivational factors of MOOC learners, and possible accreditation strategies (Liyanagunawardena, Adams, & Williams, 2013; Nguyen, Piech, Huang, & Guibas, 2014; Vihavainen, Luukkainen, & Kurhila, 2012).

The review of literature on learning CT online reveals that although the word CT might not be explicitly stated in the titles of the research papers, studies related to introductory programming are relevant to the research of CT. There are numerous opportunities online where one can learn CT concepts for the first time. Online CT learning platforms are able to collect finer grain user interaction data compared to data collected by researchers of classroom studies. Thus, these studies on online users are providing significant insight into CT learning behaviors and emerging collaborative patterns. However, studies on learners’ collaboration and learning behavior in face-to-face settings are still providing theoretical and practical understanding of what is happening online.

2.3 Face-to-Face CT Learning Opportunities

At various academic levels, different initiatives for integrating CT within existing curricula have gained footing. CT activities in the K-12 level are being integrated within various disciplinary (e.g., Mathematics, Science, English, Music and Journalism) activities. Some of these are designed to be technology free (Henderson, 2008), some require students to program games (Repenning et al., 2010) or robots, and others promote students to create their own
projects in visual programming environments (e.g. Scratch, Alice) (Weinberg, 2013). In recent years, Advanced Placement courses on Computer Science are also being considered as a fourth science course for high school graduation in some states across the US.

In higher education, some universities have been integrating CT modules throughout their undergraduate programs whereas others are offering it as a semester long course. Few of these courses are being offered from a particular discipline’s viewpoint, and others, as a general education course, designed for all non-computer science majors. The emphasis of teaching has ranged from learning CT in contexts versus learning across context, a cognitive ability versus an application of skills, as a problem solving tool or as an alternative approach to creatively expressing one’s ideas. Appendix A provides a brief summary of CT-related courses at various universities/institutions (Kafura et al., 2015). A review of the course contents offered by different universities shows conceptual commonalities among them like abstraction, procedure, algorithm, data simulation, etc. and the pedagogical approaches seem to be limited to solving problems, self-designed projects, working in groups, and learning in a flipped classroom model.

2.4 Assessment of CT

Since CT is a comparatively new concept, methods for assessing CT skills that have been published are very limited. Concept inventory and testing has been used to measure understanding of computer science concepts (Grover, 2014; Tew & Guzdial, 2011). Design based scenarios and student projects have been used to assess students’ understanding of CT by Brennan and Resnick (2012), Werner, Denner, Campe, and Kawamoto (2012). Brennan and Resnick (2012) have also conducted artifact based interviews to gain a better understanding of learners’ conceptual understanding of computational concepts. In order to evaluate students’ CT pattern (pertaining to game design), video-based prompts have been used by Marshall (2011) and a multiple choice test has been conducted by Basawapatna, Koh, Repenning, Webb, and Marshall (2011). In order to better understand learners’ reasoning processes, Aiken et al. (2012) also interviewed students. The key elements of CT learning (conditional logic, iterative and parallel thinking, and data) as well as the collaborative aspects have been studied by Moreno-León and Robles (2015), Dasgupta et al. (2016), Maloney, Peppler, Kafai, Resnick, and Rusk (2008), and Yang et al. (2015).
Review of methods for assessing CT in classroom settings indicates that it is being measured more at the individual level. Classroom studies investigating the collaborative aspects of learning CT are limited while such influences have been researched more in online learning environments.

2.5 Collaboratively Learning CT

Collaborative learning describes a situation in which particular forms of interaction among people are expected to occur, which would trigger learning mechanisms (Dillenbourg, 1999, p. 5). For a group to create knowledge, the members need to interact with each other and this interaction needs to be publicly visible (Stahl, 2010). The sociocultural perspective of learning emphasizes the importance of these interactions between group members (Robert Keith Sawyer, 2006, p. 191). Greeno et al. (1996), while studying the socio cultural perspective of learning, draw attention to social interactions that take place within a learning context. “Many learning scientist researchers argue that knowledge is first collective and external-manifest in conversation- and then becomes internalized” (Robert Keith Sawyer, 2006, p. 256).

Collaborative learning has its benefits over individual learning. It has been considered to lead to greater student success and instill deeper understanding. Collaborative learning has been effective in reducing learners’ anxiety and in helping struggling learners overcome common learning difficulties (Hmelo-Silver, 2013). Discussions between members in the collaborative groups have been considered to be beneficial for conceptual understanding (Miyake & Kirschner, 2014), knowledge construction (Weinberger & Fischer, 2006) and meaning making (J. G. Greeno, 2006). However, “for collaboration to be effective, students must explicate their thoughts; actively participate, discuss, and negotiate their views with the other students in their team; coordinate and regulate their actions between them; and share responsibility for both the learning process and the common product” (Kirschner & Erkens, 2013, p.2).

Thus, much of the success of the collaborative learning relies on the type of social interactions exhibited between group members. “If there is collaboration then social interaction can be found in it, and vice versa, if there is no social interaction then there is also no real collaboration” (Kreijns et al., 2003, p.338). The term social interaction has been used to describe the process by which one acts or reacts to people surrounding him/her (e.g., family, football
team, classroom, etc). Social interactions in collaborative learning have been characterized as describing, explaining, predicting, arguing, critiquing, evaluating, explicating, and defining, (Kreijns et al., 2003; Ohlsson, 1996). According to Bales’s framework, social interactions between group members fall under two main categories, 1) social-emotional and 2) task oriented. Social emotional interactions are observed when a group member shows solidarity, signs of tension release or agrees/disagrees. Task oriented interactions are observed in asking or giving orientation, asking or giving opinion, asking or giving suggestion. Chi’s (2009) DOLA framework categorizes social interactions based on their underlying cognitive values and interactions: active (e.g. copy, look), constructive (e.g. describing a problem in one’s own words) and interactive (e.g. when two group members jointly come to a conclusion). Thus, in order for the cognitive aspect of collaborative learning to manifest, it is also important that social interactions between group members foster affiliation, social relationships and, ultimately, a healthy community of learning (Kreijns et al., 2003).

Studies on collaborative learning have considered the individual as well as the group as the unit of analysis, and measured the proximal and distal outcomes of collaborative learning vs. individual learning (Enyedy & Stevens, 2014). Research on the community aspect of collaborative learning has focused on learners’ role in the community, expert novice relationship, help seeking behavior of learners, methods to increase participation of community members, knowledge construction by community members and exploring options to make learning communities more effective (Ludford, Cosley, Frankowski, & Terveen, 2004; Nistor, Schworm, & Werner, 2012; Schworm & Nistor, 2013; Teo & Johri, 2014). Most of the studies related to learning CT collaboratively have used data collected by online learning platforms.

Following the notion of Sawyer, this study operationalizes CT learning as “being collective and external-manifest in conversation”. It intends to understand CT learning by characterizing social interactions that occur between learners while engaged in “CT practices” (College Board & NSF, 2011). Computational concepts can be difficult to fully comprehend for novices (Lahtinen, Ala-Mutka, & Järvinen, 2005; Mayer, 1981) and collaborative learning can help such students to better understand concepts and also create a relaxed environment to learn.. Thus, all social interactions between group members may not be learning related but are still valuable in creating a positive learning environment.
2.6 Summary

Skill sets such as CT (understanding and applying computational concepts) are essential prerequisites for success in the 21st century. One can learn CT by taking a traditional course offered in a school or by self-guided learning through an online platform. CT can be practiced individually as well in a group. However, review of literature reveals that there are limited studies focusing on the collaborative process of learning CT and the ones that are available mainly address online CT learners. Thus, in order to fill this gap in the literature, this dissertation focuses on the collaborative process of learning CT. The study specifically looks into the naturally occurring social interactions between novice CT learners in two diverse settings (face-to-face vs online). The dissertation intends to illustrate the relation between self-initiated social interactions and learning CT.
CHAPTER 3 : UNDERGRADUATE STUDENTS' EXPERIENCES IN LEARNING CT FROM AN INTERDISCIPLINARY GENERAL EDUCATION COURSE

3.1 Abstract

Why would an English, an Architecture or a History major learn CT? What would they find worthwhile in learning CT? This study explores the experiences of non-computer science majors who choose to enroll in an undergraduate general education CT course. Through attention to the experiences of students with various disciplinary backgrounds, this study captures participants' points of view as they explain how collaborative interactions within the CT class influenced their learning of CT.

3.2 Introduction

Since Wing’s influential article in 2006, the term CT “has caught the attention of a broad academic community” (Grover & Pea, 2013, p.38). In her article, Wing defined “computational thinking” as “a universally applicable attitude and skill set everyone, not just computer scientists, would be eager to learn and use” (Wing, 2006, p.33). The National Academies Committee (2012) also urges the necessity for graduates to possess the ability to think computationally as one of the major skills of “21st Century”. The U.S. Bureau of Labor Statistics predicts 12 percent growth in computing related jobs from 2014 to 2024, faster than the average for all occupations and nearly three out of four jobs will require significant CS skills and knowledge (U.S. Bureau of Labor Statistics, 2016-17). As the availability of big data and integration of computing in different disciplines increases, the necessity of people in diverse disciplines with the ability to think computationally has also increased. Thus different initiatives for integrating CT within existing curricula at various academic levels have started to gain footing. Many universities are offering CT as a general education course designed for all non-computers science majors.

As institutions are offering CT courses to undergraduates from different disciplines, a typical introductory computer science course may not be suitable for this group of students. This is because these students are not the usual group of students that enroll in introductory programming courses. These students perhaps may not pursue a career in computer science. Some may or may not have strong analytical background. Thus, it is important to assess the
pedagogical approach of teaching CT to non-CS majors and investigate challenges (if any) the students experience while learning CT.

Collaborative learning has emerged as a technique that researchers have found to be generally applicable and effective for teaching Computer Science concepts (Kavitha & Ahmed, 2015; McDowell et al., 2002). Collaboration can help reduce the anxiety level of learners, improve understanding and thus create a positive atmosphere to learn CT (Hmelo-Silver, 2013). In collaborative learning particular forms of interaction among people are expected to occur, which would trigger learning mechanisms (Dillenbourg, 1999, p.5). During collaboratively learning, individuals negotiate and share meanings relevant to the problem-solving task at hand (Roschelle & Teasley, 1995, p. 70). Collaborative interactions between students majoring in different disciplines can thus promote not only a disciplinary notion of CT but also an interdisciplinary notion of CT that fosters transfer of computational concepts across disciplines (Lattuca, Voigt, & Fath, 2004).

This particular study focuses on experiences of non-computer science majors enrolled in a general education CT course. Section 3.4.3 provides more detail of the participants and the course structure. One of the main pedagogical features of the course was that students were able to collaboratively learn CT. The class was divided into semester long small groups (which was called a “cohort”) comprised of students from different majors. In this study we focus on student’s disciplinary background and how the collaborative interactions with peers may have influenced learning of CT. Stahl’s major influences of collaborative learning was used as a lens to study the collaborative features of student’s experiences (Stahl, 2010). The following section provides a brief overview of Stahl’s collaborative learning framework.

3.3 Background and Theoretical Framework

Studies focusing on collaborative learning usually consider group of learners collaboratively solving a task together. In this study, the “cohort” approach of collaboratively learning was slightly different because students in a cohort were solving the same problem individually at the same time. The cohort was designed to act as a supportive sub community within the classroom setting, taking more of naturalistic peer learning model. Thus, the study required a research framework that was flexible enough to incorporate such variations of collaborative learning.
According to Stahl, the framework is not intended to be a model of objects and processes but rather illustrates how members of small groups can engage in a cognitive activity. Hence the framework seemed suitable for this particular study which allowed the researcher to focus on students’ experiences of learning in general as well as the collaborative features emerging from their experiences.

3.3.1 Stahl’s major influences of Collaborative Learning

Various factors of a setting and the participant’s personal attributes influence collaborative learning. Group members’ prior knowledge, perspective, abilities, understanding of the task to be completed, the technology and media used, the context of interactions and, finally, the culture of the community, all play vital roles in the collaborative process. Figure 3.1 below illustrates major influences of collaborative learning as described by Stahl (2010).

![Figure 3.1 Major influences of collaborative learning (Stahl, 2010, p.6)](image)

According to Stahl’s framework, all the major influences in a collaborative learning environment interact with the sequential team interaction goal. In the context of learning CT in a classroom setting, interactions between learners, learners and the instructors, and learners and the course materials can be considered as collaborative interactions. These interactions weave all the other influences to each other. Among other influences, a student’s background, prior knowledge and perspective also impact how s/he interacts in a collaborative environment. This study focuses
on these major influences on student learning CT collaboratively, particularly on “individual” resources and experiences”. According to Stahl,

“The contributions to group discourse made by a given individual are obviously influenced by the information and knowledge that the person has—or the experiences and resources available to them. Their contributions are likely to gradually introduce this information into the group knowledge-building or problem-solving process. In fact, much of the power of collaborative learning can come from the pooling of different knowledge and alternative perspectives distributed within the group” (page 257).

Thus, in order to better understand the role disciplinary background and how it shapes a student’s experience in learning CT collaboratively, ten undergraduate students (majoring in different non-computer science) who had enrolled in a CT course were interviewed.

3.4 Research Design

3.4.1 Research Goal

The goal of this study was to investigate non-computer science majors’ experience in learning CT in a general education CT course. In order to focus on the collaborative aspects the study used the lens of Stahl’s collaborative framework (Stahl, 2010).

3.4.2 Methods

Patton (2001) suggests interviewing people to find out from them things that cannot be directly observed (p. 340). Interviews as a data collection method allows a researcher to gain better understanding of people’s experiences and perceptions. Researchers focusing on social interactions have interviewed participants to know how the participants felt doing an activity, and to reflect on their experiences (Wasserman & Faust, 1999). This study thus uses semi-structured interviews in order to gain better understanding on student’s perceptions of peer interactions in the general education CT class.

3.4.3 Data Collection

The study was conducted in Virginia Tech (VT) state university in the Southeast U.S. As part of the strategic plan the university has been offering a general education course titled
“Introduction to CT” to students from all disciplines (e.g., Political Science, English, Sociology, Animal Science, Physics, Engineering and, Business). In spring 2016, when this study was conducted the class enrolled approximately 45 students and met twice a week for 75 minutes sessions.

It is worth mentioning that the author of this dissertation study was one of the core members of the design team of the CT course of this study (when it was first offered in the fall of 2015). The researcher’s primary responsibility during the design phase was to assist in the development of course assignments and map student outcome with student performance indicators.

One of the key design aspects of this particular general education CT course was to provide students from different disciplines a course where they felt they could succeed and find the course interesting and relevant. Each computational concept was taught multiple times throughout the course. Concepts (e.g. abstraction, conditions, iteration) were introduced once in NetLogo, then again in a block based programming environment (Blockly) and again in text based programming environment (Spyder, a Python environment). For the end of semester course project, students determined their own interest-driven data sets (e.g. Facebook, Amazon, Netflix, election poling data, weather forecast data, agricultural data, sports data).

To facilitate collaboration with others in the class, the class was divided into eight cohorts. Each cohort contained students from multiple majors. Students were assigned to a cohort in the second week of the semester and the formation of the cohort lasted throughout the semester. When assigning students to cohorts, the course instructor tried to have a good mix of different disciplines and at least two female students in each cohort.

Each class day students would sit together as a cohort around a table. Students would perform all class room activities within their own cohort. The class room activities were not group assignments. Rather, students were expected to solve a problem individually but were strongly encouraged and guided in how to interact with their cohort members. A cohort “contract” was developed and signed by students of each cohort at the beginning of the semester. This contract expressed how the students in a cohort would conduct themselves and outlined their responsibilities to the cohort and to each other. Students could also collaborate virtually
using the course book, a custom-built, interactive web-based platform with embedded coding activities and real-time, shared text writing (similar to Google Docs). The assumption was that through collaborative discussions, students would gain a better understanding of the computational concepts instead of just learning individually.

In order to better understand major influences experienced by students while learning collaboratively, members of three cohorts (Table 3.1) were individually interviewed in the last two weeks of the semester. A total of 10 students were interviewed. The semi structured interviews typically lasted 15 to 40 minutes were audio recorded and transcribed. The interview protocols were informed by Stahl’s collaborative learning framework, primarily focusing on students’ background, goals, interests, and experiences while learning CT. The interview participants were voluntary. The researcher obtained Virginia Tech Human Subjects Research Approval through the Institutional Review Board (IRB) of VT before initiating the interview for the study. Consent of the participants was obtained prior to the initiation of data collection. The IRB approval for this project is IRB 14-627. This study is a part of the NSF-award 1444094 and NSF-award-1624320 investigating ‘Scaffolding Big Data for Authentic Learning of Computing”.

Table 3.1 Profile of students interviewed

<table>
<thead>
<tr>
<th>Cohort</th>
<th>No. of students interviewed</th>
<th>Total no. of students in the cohort</th>
<th>Major</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cohort Alpha</td>
<td>3 (senior:2, freshman:1)</td>
<td>5</td>
<td>History, Political Science, Architecture,</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Chemistry</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Public Relations, Housing, Theater Arts,</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Journalism</td>
</tr>
<tr>
<td>Cohort Beta</td>
<td>4 (senior:3, sophomore:1)</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>Cohort Gamma</td>
<td>3 (sophomore:2, freshman:1)</td>
<td>6</td>
<td></td>
</tr>
</tbody>
</table>

3.4.4 Data Analysis

All transcribed audio recording of individual student interviews were coded using a prior code (see Table 3.2) informed by Stahl’s collaborative learning framework. Two independent researchers coded separately for the purpose of cross-checking coding and emerging themes. Discrepancies with codes were discussed and resolved. To protect the anonymity, pseudonyms were assigned to each of the interview participant.
Table 3.2 A priori code, informed by major influences of collaborative learning by Stahl (2010, p.6)

<table>
<thead>
<tr>
<th>Codes</th>
<th>Sub-codes</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Background</td>
<td>Discipline/Major/ year in college</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Why did the student take this CT class?</td>
</tr>
<tr>
<td></td>
<td>Prior knowledge/ Abilities</td>
<td>Does the student have prior knowledge of computation concepts?</td>
</tr>
<tr>
<td></td>
<td></td>
<td>What is the student good at?</td>
</tr>
<tr>
<td>Individuals’ resources and experiences</td>
<td>Perspective</td>
<td>What is the student’s own definition of CT?</td>
</tr>
<tr>
<td></td>
<td></td>
<td>How does the student see him/herself using what s/he has learned in this class</td>
</tr>
<tr>
<td></td>
<td>Resources available</td>
<td>What resources did the student find helpful</td>
</tr>
<tr>
<td>Individual voices</td>
<td>A role the student played</td>
<td>What roles did the student play in the classroom?</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Student, mentor to other students</td>
</tr>
<tr>
<td>Culture of discourse community</td>
<td>Type of interactions between students, TA, instructor</td>
<td>What was the culture within the class?</td>
</tr>
<tr>
<td></td>
<td></td>
<td>What was the nature of interaction between students, TAs, and instructor?</td>
</tr>
<tr>
<td>Interaction context</td>
<td>Context</td>
<td>When would a student interact with another student?</td>
</tr>
<tr>
<td>Technology and media used</td>
<td>Technology and media used</td>
<td>What type of media did they use to interact with other?</td>
</tr>
</tbody>
</table>

3.5 Findings

Investigating students’ experiences through Stahl’s collaborative framework helped identify various factors that influenced learning CT by multi-disciplinary groups of undergraduate students. A brief summary of the findings are given and then explained in detail. For most of the non-science majors, learning to code was learning something conceptually different than they were used to. These students struggled to write code, finding it difficult to comprehend how precise one had to be. Nevertheless, most of the students could comprehend the value of learning CT and its applicability to their own discipline. Most of them found the course instructor,
teaching assistants and the long term cohort model to be valuable resources in learning CT. Asking and explaining problems to cohort members were not only useful to advance through a problem but also valuable to one’s own learning process. Discussing problems with students coming from different disciplines allowed them to see how differently students perceive and explain a problem. Students coming with a science background could recognize the struggle non-science majors faced while learning to code. Providing ample time for students to actively learn CT during class time, acknowledging that students will make mistakes, allowing students to casually communicate with each other, and the overall structure of the course were few salient features of the CT course that students brought up during their interviews. The following sections summarize student’s personal CT learning experiences according to the major influences highlighted in Stalh’s framework.

3.5.1 Background of students, Prior Knowledge and Ability of students

Participants of this study mostly came from a liberal arts discipline (e.g. English, History, Public Relations, International Studies etc.) rather than Engineering, Science or Mathematics. There were few students who were majoring in architecture or in an area of science. Most of the students taking this CT course had never taken a course in coding or Computer Science before. Few students, however, had taken courses where they had to use computer applications. One student had taken computer science in high school and an introductory to Python course in college.

All students of this study were taking the CT course to fulfill a general education requirement which is referred as an “Area 5” course at VT. According the degree guidelines of the institution, an Area 5 course would allow a student to:

1. Increase basic competence in quantitative reasoning and problem solving, starting at an appropriate entry level;

2. Understand some fundamental principles of reasoning that are involved in mathematics or logic;

3. Understand quantitative and symbolic reasoning through the study of significant applications of mathematical sciences.
A course in calculus was the most common class taken by student to satisfy this Area 5 requirement prior to the offering of this CT course. Most of the participants found math difficult and the option of taking something different encouraged them to take the CT class. The students expressed their anxiety in math in the following excerpts.

“I'm not really good with math ... My brain's more of the English and history side versus science and math. So it's been good for the math section to fill. This is a good option for that section for people who aren't good at math, honestly.” [English Major]

“Math is not my strong suit so I decided to take this class because I will probably use a different perspective and it will be little bit easier than calculus.” [International Studies Major]

Apart from finding math difficult, some students also could not relate the use of mathematics particularly calculus, algebra in their discipline or future profession. On the other hand, learning something to do with big data, coding, computers sounded interesting and valuable.

“I think it's definitely better than taking a standard math course. Because I think in this course, it's brought out more of how this impacts the world. Because I mean, right now, it just seems like math, algebra, and physics doesn’t connect. How am I going to apply this to the real world? But this actually shows us how we can use this data and how we can use it to represent our ideas.” [Theater Arts Major]

“I'm taking it to really learn more about the big data side of computational science.” [Science Major]

Also, having someone in the family specializing in the area of computer science motivated a student to take this course.

My brother is a he's a computer engineer, working in New York. And I've always been kind of interested in what he does, so I thought it would be a good way to get a little insight on to the world of computer programing [Architecture student]
3.5.2 Perspective

Students from different disciplines elaborated on their perspective of CT and how it may be useful in their own professions.

“This [CT] is pretty useful, this information, and regardless of what job you're going to have in the future, and I feel like this might become more useful, because you use computers all the time.” [International Studies Major]

CT as a way of thinking, which is breaking a problems to their most basic level, using the power of computation to solve a problem faster and more efficiently and considering the results and it’s social implications.

“Well, I think it's trying to come to a creative solution using analytical methods that, you know ... A computer is not a smart device, it just follows orders, so I think computational thinking, like learning the concepts of it allows you to think almost like a computer "thinks." And that is a very basic level, basic way to understand complex ideas.” [Architecture Major]

“I guess computational thinking would be being the brain behind the computer which is doing the action. The way we were explained to it was the computer is only as smart as the user. It can do the actions extremely fast, and it can do a lot of it at once in an extremely fast manner. So the creativity behind doing that, finding the information and sorting it and giving it to whoever wants to see it is the computational thinking part.” [English Major]

“Computational thinking in my own words would probably be using a computer to extend your thinking, to be able to analyze something farther than what your brain is able to do. A computer can detect patterns like there's no tomorrow, and it's something that your brain might not be able to correlate because there's too disparity between it. Being able to use a computer to think almost better than what we're able to just on our own.” [Science Major]

Students majoring in different disciplines were able to relate the use of CT to their fields. For some students, the final project of the CT class helped them to make this connection. For example, a student in journalism was able to describe how big data is being used to analyze and predict crime-related behaviors. A student in Public Relations understood that s/he might need to apply his/her computation skill while developing websites. The student majoring in chemistry
explained how computational analytics help chemists to discover new drugs. The following excerpts are taken from the conversation between the journalism, architecture and chemistry majors respectively

“It's sort of on the cusp of journalism is now, they're going through very large things of data. Most recently, the Chicago red light thing that was going on with red light cameras in Chicago. Somebody had to go through all 4 million instances of red light tickets. And you could go through them manually, or you can have a programmer go through all of it. If you can do that as a journalist it's getting more and more to where you kind of need to have a niche position. Almost as if the job has never been there before, like Washington Post or something like that. Somebody's going through all that data to create the graphs that end up on the news.”

[Journalism Major]

“I'll probably be able to use it ... The coding, the different rules for coding. I'm actually working on a website in another of my classes so it's helpful to know how coding works. Then also just how we're working on the visualizations for the final project right now, I think that's going to be pretty helpful in the future to be able to analyze data in my PR I'll probably be able to use that.”

[Architecture Major]

“Computers use a lot in chemistry, as with collecting of data and then analyzing that data and then being able to tell not only the scientific community, but the rest of the world as well. Look, we came up with this new process that does this, and it's this great new drug or it's this new chemical compound. Being able to use computers in that sense, to be able to analyze that data and be able to effectively then distribute it and everything would be a very big help later down the line.”

[Science Major]

However, some students still had a very rudimentary perception on CT and were not able elaborate on the applicability of CT in their own field. For example, in the following quote, a History major gives a very brief definition of CT

“I would just say using computers to analyze data.”

[History Major]

The profiles of the interviewed participants revealed that students who were more advanced in the study of their discipline were able to elaborate on the connection of CT to their own
discipline while freshman and sophomore were not clear in making the connection. This difference could be explained by the fact that the freshman and sophomore students were new to their own disciplines and their disciplinary perspectives were not fully formed.

3.5.3 Culture of the Discourse Community

For some students the discourse culture of CT class formed by the introductory emails and the reddit posting of the CT course. These communications described the CT course as a general educations course in which students coming from different disciplines would be able to do a final project with data sets from their own discipline. This option of using disciplinary data sets gave students coming into the course a sense of connection to their own discipline.

“I got an email about this class and it sounded really fun, and instead of just fulfilling an Area 5, it was like, "You get to do a project that is in your own major," and all this kind of stuff you get to do is very individual-based. I really liked the appeal of that so I ended up switching into this one. I feel like it's good skills to have in an increasingly technological society.” [English Major]

Some students got feedback from students who had already taken the course. Former students suggested the CT course to be a good course to take instead of one in mathematics.

“I actually learned about it because my RA was in the class. And she told me about it because I just didn't really want to take calculus.” [Political Science Major]

“My friend, who was a freshman and took this course last year... and she said it was a good course, so I decided to take it. [Theater arts Major]

3.5.4 Student Voices (Role of a student)

Students can take on various roles in a collaborative environment. Teamworker, coordinator, implementer are just few common roles in collaborative literature (Belbin, 2004). However, in this particular study the roles students assumed were slightly different, more oriented towards peer learning. This is because students were not solving a single problem collaboratively. Each student was solving the same number of problems individually. If they needed help they could voluntarily ask help from their group members. The main roles students conveyed in their interviews were asking a question, responding to a question and taking the role of a mentor.
Most of the students interviewed explained their role in the cohort as being both a tutor and a tutee. Students would work on the problems by themselves and if they got stuck, they would ask help from a cohort member.

“During the class we would work on our own and if we had questions we would ask each other sometimes I would ask, sometimes someone else would ask me” [Architecture Major]

One particular student assumed more of a mentor role. This student had prior programming experience. He was also willing to help others. As a result he was seen helping out most of the other students in his cohort. His cohort members also found his presence to be reassuring. In the following quote, a Liberal Arts major describes his role in the cohort.

“I really liked our group. We would just work on our stuff and mostly ask questions, we'd go to Jonathan because he's the only one who has any CS background. So he knows everything compared to us. We would ask him a lot of questions. He's a super helpful guy and super talkative…I had lots of learning opportunities, from Jonathan especially and then from other people much less so. I think also another reason that everybody was here all the time is because we had Jonathan. He had that kind of background experience. Even if nobody else was getting it, Jonathan gets it. I think we always felt safe to come to class” [Liberal Arts Major]

This particular cohort mentor was a science major with prior experience in programming. In the following quote he describes his first couple of weeks of CT mentoring experience.

“I wasn't really prepared to take on this role, so I wasn't thinking I need to be a leader. It was sort of like oh, I know about this stuff and you guys don't really know all that much about this stuff, I guess I should help you out, and it was sort of a last minute thing. It kind of shook me for the first couple of weeks. I was like these people really need my help.” [Science Major]

Although some students in a class may have more experience with the subject matter being taught, coming into the class a student may not be prepared to take on a mentoring role. In the above excerpt, the science major expresses his dilemma. However, he realizes that students learning to code for the first time needed help that he could provide.

In the quote below, a Science Major notices the struggle students from different disciplines experience while learning to code. He recognizes that his background in science
helped him to learn to code more naturally whereas it was much more difficult for students with a non-science background. He identified that his thought process was more aligned with the process of learning how to code than other members of his cohort who had a liberal arts or architecture background.

“It was interesting because I've never been really in a peer mentorship position before where people were like oh, let's talk to Jonathan about this problem. It was definitely an experience for me, being able to help these people who have never taken CS before in their entire life. I was traditionally trained with hard coding like this is how you code it. They had a much different take on it, so it was very interesting.…

I am a science background. I've always been a science background and I'll always be a science background. I like science. My mind is very analytical, so it was really easy for me to learn coding. This is how logical it is. It goes step by step by step, where some of others take was not similar... I don't get that. Maybe because this was their first take on coding, and they're doing something rather large, rather quickly. It was interesting having to help walk them through this large jump into this project. They all got it in their own time. Some took slightly longer than others.” [Science Major]

3.5.5 Resources Available

When asked about what resources were available to learning CT, the students referred to the course instructor, TAs, the structure of the course, online course book, and the formation of cohorts to be beneficial.

“The instructor was super helpful, without just giving you the answer, which is good…And the instructor has been really great to help you figure it (the problem) out on your own, so you can understand what's going on. And that's been really good.” [Architecture Major]

“There weren't always things that the cohort members could answer, but working through it with the instructor and the TA, they just have a way of explaining it that makes sense to me.”[English Major]

However, one interviewed student did not find the TA to be well prepared and helpful.
“Our TA wasn't super helpful in anything. He was a lot of, ‘I'm not really sure’. Then he wouldn't even check on it. He would just tell us he didn't really know. It wasn't very helpful.”

3.5.6 Context of Interactions

The context in which students interacted with others, a course instructor, another peer or the course material also influences the students’ experience. The following three sections explain (i) when students interacted with instructors or TAs, (ii) when they interacted with peers and (iii) when they interacted with technology related resources.

3.5.6.1 Interaction between student-Instructor/TA

The students in this study valued their interactions with the instructor and TAs. Whenever a student had a problem or needed clarification, s/he could ask for help from the instructor or TAs in class or through email. The students of the CT course found the instructors and TAs in general to be approachable and caring.

“The instructor was always found walking around to ask if we need any help, or just to check in and to see what we're working on and to check, make sure we're on track”

[Architecture Major]

“A few times I had this very, very, very simple problems with my program or something like that, and I would email TA and he'd get right back to me.” [Theater Arts Major]

An English major found it comforting that the course instructor acknowledged that students might make mistakes in the class.

“Being able to learn in this class and make mistakes and it not be the end of the world for that, means everything to me for this class. Just because making mistakes is a huge part of the learning process. Increasingly, making mistakes is not allowed in our culture anymore and that just piles the stress on.” [English Major]

3.5.6.2 Peer interactions

The students believed the cohort model to be beneficial to learning. The students appreciated the in-class time allowed to learn CT with cohort members.
“I think it's because we were provided ample time every single class to work as a group… I think that was probably the biggest facilitator of the group interaction was just being in the same place at the same time at a time that was convenient for everyone, because we were planning on being there for class.” [English Major]

The Journalism Major appreciated the cohort as a natural conversational mode with cohort members.

“I guess I was really fond of the way we could just casually bring up the conversation instead of formally raising your hand, and the class would stop ... I just felt like it was more natural conversation.” [Journalism Major]

One of the main reasons why students found cohort model to be helpful is to have someone at the same level of learning explain a problem.

“Mostly because we're all kind of on similar levels of understanding, my cohort members and I, so when they explain it in their own words, it's just easier for me to understand” [History Major]

According to the History major it is helpful when a learner with the same level of understanding explains a problem in their own words.

“And for me, I personally would rather hear a student’s point of view on a problem because I probably relate to them more than an instructor. Usually the student will give some sort of example to coincide with the problem. So, I guess it's not as hard to grasp. While the instructor will use more of the technical terms which I probably don't know so, it's harder for me to grasp the concept.”[Political Science Major]

A Political Science major preferred an explanation from a peer student rather than the instructor. This particular student found the use of technical terms by the instructor to be difficult to interpret. In the excerpt bellow a student majoring in Theater Arts describes that at times when he was burdened thinking over a problem another cohort member would explain the problem in more basic terms which was helpful.

“Like if I was more confused, if I was overthinking a problem in something, my cohort members would boiled it down to more bare basics. That was helpful.” [Theater Arts Major]
In the two excerpts below a student majoring in English and another majoring Architecture describe that explaining a problem to another cohort member can be helpful for one’s own understanding.

“A lot of times we would just say the things we were doing, especially in Blockly, as we were doing them. I've definitely learned, not just in this class but overall as a psychology thing, being able to explain back something that you've been told is a really good way of gauging understanding. It's always funny when you ask a question to our group because you'll get some really interesting answers. Part of them will be funny and part of them will be legitimate, but we all shared the struggle I guess.” [English Major]

“Others helping me helps me learn it through, like on a one-to-one scale with a person my age. It's kind of relatable. So you're learning with someone who's learning with you, and if they know something, it's like oh, I can learn it too. And if I know something they don't, they can learn it too, and we're just helping each other grow. Of course, if I help him with something, and I can verbally communicate it with him that means that I'm understanding it more, and that's definitely a learning skill.” [Architecture Major]

The Architecture major in the above excerpt elaborates on the aspect of enhancing self-confidence. Seeing someone with similar experience in learning CT able to solve a problem helped him to believe that he could also solve the problem.

Few Liberal Arts students found learning CT to be a different type of learning then they were used to.

"I don't get this. I'm not used to having to be so precise and so logical about everything."
[Political Science Major]

When the student was asked to elaborate on what he meant by “being precise and logical” the Political Science Major elaborated,

“There is more than one way to write the same thing. Even if your sentence is not fully formed others can figure out what you want to say. But with computers, it has to be picture perfect, or else it doesn’t work.”
“I'm not really good with ... My brain's more of the English and history side versus science and math” [English Major]

Whereas for the English major, learning to code was similar to learning a new language.

“As Area 5 there are courses in math, statistics, there's some philosophy ones in there. I tried that one time and it was hideously horrible for me. I'm a very language kind of person and so I feel like I'm learning another language (in the CT course), which is really cool.” [English Major]

Students appreciated the exchange of ideas with cohort members and having students coming from different disciplines in their cohort.

“Students coming from different discipline had a different take on how they explained a problem or solution of a problem and that was nice”. [Political Science Major]

Students appreciate the long lasting cohort model for the community aspect it provided for the students.

“Since we're able to all talk about it, like we're comfortable sharing our problems or our complications with each other, it was easy to learn like that, because it was just like you weren't afraid to ask for help, and sometimes, your cohort members would explain something a different way, and then it would help you understand it.” [History Major]

The students mentioned that it was important not only to talk about and solve problems in a cohort but to get to know each other, attend class regularly, tell a joke and laugh about things happening in each other’s lives. These non-academic conversations made them feel comfortable to ask questions to each other and probably also kept the cohort functional.

“Building that sense of community, just friendship and working relationships with the people in my cohort was a huge part I think for us of how we got along so well, just little things here and there. We weren't just coming to do the group work, we were all getting to know each other and I think that really helped.” [English Major]
3.5.6.3 Student- technology

The students valued how the class was structured in terms of content: gradually moving from visual programing to text based programming, having ample time to solve problems in class, and the way the class was set up: seating arrangements, round tables, forming cohorts.

“I feel that the way they taught it was helpful. When we started with NetLogo, then we went to Blockly, and we went to Python after that. I feel that was a pretty smooth transition. Because NetLogo, it starts with dealing with models and what the code can do in terms of what it shows it's from the data. Blockly was good because it's definitely helpful in learning what words go in which places and learning the format, and then Python of course, just combines all that into creating what you want with the words you want and the models you want.” [Architecture Major]

In terms of technology, the students appreciated the course material offered in one place, an online book. They did their work in it, referring to reading material while they were solving problems.

“Having everything at one place was very helpful. You could easily read up if you had any confusion during solving a problem.” [Public Relations Major]

Students in the CT class did not only communicate with their cohort members within class but also on Facebook. They had a Facebook group where they communicated, shared their experiences, problems they were having with home works and help one another outside the class room.

“We have a Facebook message for if we're doing homework or something and we need help that we're all linked into so we all see it. We can then message, "Oh, well I see your code. You really need to put a parenthesis here. You're not counting this right," or something like that. We can all help each other, not while we're even just in the classroom. We can meet outside the classroom.”[Science Major]
3.6 Discussion

In studying small group cognition, Stahl suggested focusing on the role, interactions and discourse between group members (Stahl, 2013). In this study, students’ experiences while learning CT were also influenced by other factors that contribute to small group cognition. Students’ role and the discourse within the CT class were shaped by the resources that were available, and the interactions student had with their peers, the instructor and the TAs.

The findings of study reveal that most of the students taking this CT class did not have prior experience in learning computational concepts and were enrolled in College of Liberal Arts, College of Architecture or College of Science. Figure 3.2 below maps Stahl’s collaborative framework with factors influencing the experiences of non-computer science majors learning of CT at the undergraduate level. Only a few students did have some sort of prior programming experience. Most of the students were taking this course as alternative to taking a math course (calculus). Thus, student’s discipline and prior experience in computing were considered as the background of the participants. The discourse in the small groups as well in the class was less formal. Students could casually start talking to one another.
Figure 3.2 Factors influencing non-computer science majors experiences in learning CT within small groups

In terms of available resources, students appreciated the support provided by the instructor and the in-class teaching assistants. They found having a semester long small groups (cohort model) helpful to learning CT. Students characterized their role in the small groups as a tutee (mostly asking questions) or as a tutor (providing explanations). Some students also characterized their role as mentors. Students found it beneficial to explain and be able to see how others solved a problem. The students found it comforting that they were able to work on a data set from their own discipline. The disciplinary data sets allowed the students to bring in their own disciplinary knowledge and combine what they had learned from the CT class.

In terms of experiences, students referred learning CT to be different than typical type of learning they were commonly used to. Breaking down problems to its very basic level so that a computer can understand seemed difficult for them to comprehend. For some students learning to code was like learning another language. However, for a student majoring in science, learning CT was not that much different.
In terms of outcome, students referred to being able to comprehend what CT is, the applicability of CT within one’s own discipline, the ability to write a computer program and having a sense of community within the class.

3.6.1 Implications for practice

Students with a non-science background can find learning CT difficult. On the other hand, the process of learning CT is more intuitive and comes naturally to the students coming from a science background with prior programming experience. Using discipline specific datasets can be helpful to keep students interested and motivated to learn CT. Forming semester long groups/cohorts can be beneficial for novice learners. These small groups provide students a sense of belonging, a place to share thoughts and anxieties. Students find it is easier to naturally ask questions to group member than help from instructors. An explanation coming from a peer who is at the same learning level is easier to understand and relate to. Educators should also keep in mind that some students may struggle and need extensive support.

3.6.2 Limitations and future directions

Students who participated in this study were self-selected. For this reason, it is likely that most of them had a positive attitude towards the CT class. Also, most of the students interviewed in this study were taking this particular CT course to avoid taking a course in ‘math’. This could have influenced the findings of this study. There were disciplines from which students were enrolled in the course but were not interviewed. Interviewing students from a larger number disciplines could have allowed the researcher to gain a broader perspective. A future direction of this work could be to investigate the similarities and dissimilarities within student’s disciplinary backgrounds that support or hinder learning of CT for undergraduate students.

3.7 Conclusion

Findings of this study suggest that novice CT learners from different disciplines had varied experiences in learning CT. The students valued the cohort model of learning CT collaboratively. Disciplinary data sets allowed students to combine their disciplinary knowledge with computational skills learned from the CT class. The students also valued having a learning environment where the faculty acknowledged that the students might find learning CT difficult and provided extended support.
CHAPTER 4 : COLLABORATIVE MOMENTS OF LEARNING CT: EVALUATING NOVICE UNDERGRADUATE STUDENT’S SOCIAL INTERACTIONS IN SMALL GROUPS

4.1 Abstract

Background Computational language is dawning as the lingua franca of this century. Abundance of data of various types from all disciplines and availability of tools of computation and algorithms have made it easier to explore such datasets. However, there is a need for professionals with computational skills to provide disciplinary insight and guide such computational endeavors. One of the central views of CT is the ability to interpret and relate CT within one’s own discipline. Thus, as a 21st Century skill set it is necessary for graduates from all disciplines to be equipped with computational thinking skills. Collaborative learning has emerged as a productive approach for learning Computer Science. Collaboration can help reduce anxiety level of learners, improve understanding and thus create a positive atmosphere of learning. Group of learners with diverse experiences, values and knowledge can often be more effective in helping novice learners comprehend concepts of Computer Science. However, there is limited research focusing on how natural collaborative interactions among learners from different undergraduate disciplines manifest during learning of computational concepts.

Purpose This study investigates interaction between members in small interdisciplinary student groups/cohorts in an undergraduate CT class at Virginia Tech University largely serving students from non-science and non-engineering majors (e.g., Political Science, English, Sociology, and Animal Science, etc.). The underlying assumption being that students in a cohort from different disciplines will bring diverse perspective to the groups, socially interact with each other and in turn create situations where two or more student learn or attempt to learn CT together. This study attempts to provide naturalistic accounts of such social interactions between students collaboratively learning CT in an undergraduate classroom setting.

Method: This study used ethnographically-informed qualitative data collection methods (observations and interviews). Data collection was conducted in one academic term. Three student cohorts working during class time were video recorded during three class sessions. Each video is approximately 20 minutes in duration. Each cohort comprised 5 to 6 students. All transcribed
recordings were coded by two independent coders. First, in each transcript, episodes of social interactions were identified. Second, each episode was then categorized based on the adapted version of Chi’s Differentiating Overt Learning Activities (DOLA) framework.

**Results:** Analysis of interactions between cohort members revealed that undergraduate students coming from non-engineering or science disciplines found it difficult to transform problems into computationally solvable parts. Even though most of the cohort interactions were active in nature, the students struggled especially when their peers themselves were novices. The students who lacked knowledge of computational concepts found it hard to contribute to a discussion. Instead they chose to show their work to others or ask a direction question. Students who had a better hold on a concept were constructive in nature. While being constructive, they would elaborate on the topic, make connections with previous problems and were self-aware of their own understanding. Although different cohort functioned at different levels, in general the cohort was considered a safe place for students to share anxiety, express frustrations, ask questions, receive explanations and locate technical resources. Attendance, socialization, prior programming experience, and willingness to help others played a crucial role in differentiating the type of interactions students had while collaboratively learning CT.

4.2 Introduction

The notion of CT has been proposed as a core set of concepts for teaching CS by Wing (2014, 2011, 2006); Bundy (2007). According to Wing (2011) “if computational thinking will be used everywhere, then it will touch everyone directly or indirectly” (p.3720). Bundy (2007) phrased the universal applicability of CT as “Computational thinking is influencing research in nearly all disciplines, both the sciences and the humanities” (p. 67). According to CSTA and ISTE (2011), in order for students to realize CT is important for them to recognize and use computational concept within their own discipline/interest and realize the variety of problems and scenarios CT can be used in (CSTA & ISTE, 2011).

Following the notion of CT is everywhere and for everyone (Brennan et al., 2010; Wing, 2011) this study is situated in a CT class where students majoring in different disciplines (e.g. Political Science, English, Sociology, Animal Science etc.). One of the key learning objectives of
this course was to improve students CT skills. In order to better understand how students from different disciplinary backgrounds learn CT at the undergraduate level this study focuses on student’s interactions in small interdisciplinary groups within this classroom setting. This part of the study focuses on answering the following research question:

i. How do novice CT learners naturally interact with group members while learning CT in a classroom setting?

The study provides naturalistic accounts of non-computer science majors attempting to learn CT. Chi’s DOLA framework has been used to better understand the nature of such accounts and how these relate to learning of CT. The following section provides a brief background on collaboratively learning CT and the DOLA framework.

4.3 Background and Theoretical Framework

4.3.1 Collaboratively Learning CT

Collaborative learning has emerged as a technique that researchers have found to be generally applicable and effective for teaching Computer Science concepts (Kavitha & Ahmed, 2015; McDowell et al., 2002). Most novice learners of CT, particularly those with a non-science background can find it difficult to learn how to code for the first time. This is because they lack the mental model and domain specific knowledge of computing (Mayer, 1981; Sweller & Chandler, 1994). A collaborative learning environment, where students in small groups can freely interact with the group members can create a positive and successful CT learning experience. Interactions in the form of explaining, predicting, arguing, critiquing, evaluating, and defining can help novice CT learners to better understand computational concepts (Kreijns et al., 2003). However, just forming students into groups does not necessarily suggest that students will collaborate with each other. The social aspects of learning in collaboration is unpredictable, happens naturally and depends on the behavior of both students and instructors (R. Keith Sawyer, 2013). Some behaviors can lead to increased learning, whereas others may not contribute to learning at all. This study focuses on the overt behaviors of group members in order to identify collaborative moments of learning CT. To evaluate such collaborative moments, Chi (2009)’s DOLA framework has been used. The following section provides a brief overview of the framework.
4.3.2 Differentiated Overt Learning Activities (DOLA)

Chi (2009)’s DOLA framework provides an evidence based taxonomy to categorize student engagement while learning. A student is active when s/he is doing something (physically or verbally). An example of an active learning would be when a student paraphrases a text or when a student uses a simulation software to manipulate an existing scenario. A student is constructively engaged when s/he goes beyond what s/he was initially provided. Activities such as self-explanations where a student explains what a text sentence or a solution step means to others in his own words is considered constructive. Students are interactively engaged when they can explain the subject matter in detail and each partner considers the contribution of the other. An example of an interactive activity would be when a student asks an instructor for help and the instructor provides feedback which leads to a more extended dialogue on the issue.

Chi’s DOLA framework suggests that some behaviors are better in engaging students in deeper learning than others. According to the author, an interactive activity involves higher cognitive process than constructive, a constructive activity is higher cognitively than active activity. Figure 4.1 illustrates the level of learning ascertained by the DOLA framework.

![Figure 4.1 Different types of overt learning activities in the DOLA framework (Chi, 2009)](image-url)
Table 4.1 illustrates different activities that constitute as active, constructive or interactive and cognitive process associated with it. In this study the term ‘overt activity’, ‘social interaction’, and ‘exhibited behavior’ are used interchangeably. These terms refer to an activity that is externally observable (seen or heard) during collaboration.

Table 4.1 Different activities that constitute as active, constructive or interactive and cognitive process associated with it

<table>
<thead>
<tr>
<th>Characteristics</th>
<th>Active</th>
<th>Constructive</th>
<th>Interactive</th>
</tr>
</thead>
<tbody>
<tr>
<td>Doing something Physically</td>
<td>Producing outputs that contain ideas that go beyond the presented information</td>
<td>Dialoguing substantively on the same topic, and not ignoring a partner’s contributions</td>
<td></td>
</tr>
<tr>
<td>Look, gaze, or fixate Underline or highlight</td>
<td>Explain or elaborate Justify or provide reasons Connect or link</td>
<td>Respond to scaffoldings</td>
<td></td>
</tr>
<tr>
<td>Gesture or point</td>
<td>Reflect, or self-monitor</td>
<td>Revise errors from feedback</td>
<td></td>
</tr>
<tr>
<td>Paraphrase Repeat</td>
<td>Plan and predict outcomes</td>
<td>Build on partner’s contribution: argue, defend, confront or challenge</td>
<td></td>
</tr>
<tr>
<td>Searching Repeating sentences verbatim from a question</td>
<td>Self-explanations are meaningful elaborations that go beyond and are not explicitly presented in the learning material</td>
<td>Interacting with an expert in instructional dialogues (explaining, providing corrective feedback, guiding the learner (e.g. asking questions, giving hints, initiating steps) Interacting with a peer where both peers make substantive contributions to the topic or concept under discussion (e.g. building on each other’s contribution, defending , arguing a position, challenging and criticizing each other</td>
<td></td>
</tr>
<tr>
<td>Copying problem solutions steps from another student</td>
<td>Asking questions</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Selecting from a menu of choices</td>
<td>Compare and contrast cases</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reflect or monitor one’s own understanding and self-regulatory activities</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cognitive processes Attending Processes Activate existing knowledge</td>
<td>Creating Processes Infer new knowledge Integrate new information</td>
<td>Jointly Creating Processes Creating processes that</td>
<td></td>
</tr>
<tr>
<td>Assimilate, encode, or store new information</td>
<td>with existing knowledge</td>
<td>incorporate a partner’s contributions</td>
<td></td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------------------</td>
<td>---------------------------------------</td>
<td></td>
</tr>
<tr>
<td>Search existing knowledge</td>
<td>Organize own knowledge</td>
<td>for coherence</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Repair own faulty</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>knowledge</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Restructure own knowledge</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The DOLA framework has previously been used to evaluate student engagement in physics, mathematics, bridge design, evolutionary biology, human circulatory system, introductory materials science and engineering classrooms (Michelene Chi, 2009; Michelene Chi & Wylie, 2014; Menekse, Stump, Krause, & Chi, 2013). The framework has been mostly used in quantitative studies and used to evaluate student interactions between pairs (Michelene Chi, 2009; Michelene Chi & Wylie, 2014; Menekse et al., 2013).

In this qualitative study the DOLA framework was used to develop a priori code to evaluate collaborative CT learning behaviors. Using the DOLA framework this study will illustrate collaborative moments that foster CT learning in novice learners.

4.4 Research Design

4.4.1 Research Goal

The goals of this study was to answer the following overarching research question

- How do novice CT learners naturally interact with group members while learning CT in a classroom setting?

In order to answer this research question this study evaluated naturally occurring social interactions amongst groups of college students and evaluates these interactions using Chi’s DOLA framework. The DOLA framework allowed us to categorize (as active, constructive and interactive) interactions based on their underlying cognitive process and suggest their contribution towards learning CT collaboratively. In this study social interactions between students are considered as naturally occurring because the students were not obliged to interact with each other. The interactions between students happened spontaneously.
4.4.2 Method

This study employs ethnographically-informed data collection methods (e.g., field notes, observation, interview, etc.). Qualitative ethnographic methods have been suggested by Sawyer to be appropriate when the researcher considers social interactions as the mediating mechanism for collaborations contributing to learning (R. Keith Sawyer, 2013). In this study the collaborative process of learning CT was analyzed by focusing on the social interactions between group members. Thus, according to Sawyer’s suggestion, qualitative approach of investigation seemed best suitable for this study.

To better understand the social interactions between group members learning CT in naturalistic settings (in this particular study the naturalist setting is the classroom), focused participant observation (Spradley & Baker, 1980) was conducted. The focus of observation was conducted in the time frame during class when students interacted with each other. Studies investigating social interaction (Kelly & Crawford, 1996; Webb, Ing, Kersting, & Nemer, 2006) that used observations as their main data collection strategy usually consider observation data to be the audio or video recording of students working together. Thus, students working in collaboration in the CT class was video and audio recorded. Along with the recordings, the researchers also took observation notes during and after observing students’ group work, collected student generated computer log data, and final projects of all students.

In a qualitative research study, a researcher can take one of various roles in the research settings (as summarized in Table 4.2). The role can be defined by how much the participants know about the researcher’s object and how involved the researcher is in the research settings (Creswell, 2008). According to Creswell, the researcher’s role can be varied: it can be completely concealed from participants, or the role of the researcher is known, or the researcher him/herself is participating in the setting and the role of observation is secondary or the researcher observers without participating.
Table 4.2 Different types of roles of the observer (Creswell, 2008)

<table>
<thead>
<tr>
<th>Different types of observers</th>
<th>Description of role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Complete participant</td>
<td>Conceals role and participates in setting</td>
</tr>
<tr>
<td>Observer as participant</td>
<td>Reveals role and participates in setting</td>
</tr>
<tr>
<td>A participant as observer</td>
<td>Secondary role is to observe and participates in setting</td>
</tr>
<tr>
<td>Complete observer</td>
<td>Reveals role but does not participate in setting</td>
</tr>
</tbody>
</table>

In this study, the lead researcher’s role was revealed to the participants. However, her sole objective in the setting was to observe participants. Thus the lead researcher’s role can be characterized as ‘complete observer’.

4.4.3 Research Participants

The study was conducted in a large regional state university (Virginia Tech) in the Southeast U.S. As part of its strategic plan the university has been offering a general education course titled “Introduction to Computational Thinking” to students from all disciplines (e.g., Political Science, English, Sociology, Animal Science, Physics, Engineering and, Business). In fall 2015, when this study was conducted the class enrolled approximately 45 students and met twice a week for 75 minutes sessions.

One of the key design aspects of this particular general education CT course was to provide students from different disciplines a learning experience where they felt they could succeed and find the course content interesting and relevant. Each computational concept was taught multiple times throughout the course. Concepts (such as abstraction, algorithms) were introduced once in NetLogo, then again in a block based programming environment (Blockly) and again in text based programming environment (Spyder) for a standard programming language (Python). For the end of semester course project, students determined their own interest driven data sets (e.g. Facebook, Amazon, Netflix, election poling data, weather forecast data, agricultural data, sports data).
To facilitate collaboration among students, the class was divided into eight groups. Each group (which was called a “cohort”) contained students from multiple majors. Students were assigned to a cohort in the second week of the semester and the formation of the cohort lasted till the end of the semester. While assigning students to cohorts, the course instructor tried to have a good mix of different disciplines and at least two female students in each cohort.

![Image](image1.jpg)

Figure 4.2 Students of the CT class solving problems in a cohort during class

Each class day students would come and sit together as a cohort around a table (Figure 4.2). Students would perform all class room activities with their own cohorts. The class room activities were not group assignments. Rather, students were expected to solve a problem individually but were encouraged to interact with their cohort members as needed. A cohort contract was developed and signed by students of each cohort at the beginning of the semester. Students could also collaborate virtually using the course book, a custom-built, interactive web-based platform with embedded coding activities and real-time, shared text writing (similar to Google Docs). The assumption was that through collaborative discussions, students would gain a better understanding of the computational concepts instead of just learning individually. Table 4.3 summarizes the research participant’s gender, and major of students of each cohorts observed. The study uses the pseudonyms Alpha, Beta and Gamma to refer to each cohort.
Table 4.3 Participant’s cohort, gender, and major in the CT course

<table>
<thead>
<tr>
<th>Cohort</th>
<th>Student No.</th>
<th>Gender</th>
<th>Major</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alpha</td>
<td>Student 1</td>
<td>Female</td>
<td>Theatre Arts</td>
</tr>
<tr>
<td>Alpha</td>
<td>Student 2</td>
<td>Female</td>
<td>History</td>
</tr>
<tr>
<td>Alpha</td>
<td>Student 3</td>
<td>Male</td>
<td>International Studies</td>
</tr>
<tr>
<td>Alpha</td>
<td>Student 4</td>
<td>Male</td>
<td>Architecture</td>
</tr>
<tr>
<td>Alpha</td>
<td>Student 5</td>
<td>Female</td>
<td>Political Science</td>
</tr>
<tr>
<td>Beta</td>
<td>Student 1</td>
<td>Female</td>
<td>Political Science</td>
</tr>
<tr>
<td>Beta</td>
<td>Student 2</td>
<td>Female</td>
<td>International Studies</td>
</tr>
<tr>
<td>Beta</td>
<td>Student 3</td>
<td>Female</td>
<td>English</td>
</tr>
<tr>
<td>Beta</td>
<td>Student 4</td>
<td>Male</td>
<td>Multimedia Journalism</td>
</tr>
<tr>
<td>Beta</td>
<td>Student 5</td>
<td>Male</td>
<td>Chemistry</td>
</tr>
<tr>
<td>Beta</td>
<td>Student 6</td>
<td>Male</td>
<td>Architecture</td>
</tr>
<tr>
<td>Gamma</td>
<td>Student 1</td>
<td>Female</td>
<td>Public Relations</td>
</tr>
<tr>
<td>Gamma</td>
<td>Student 2</td>
<td>Female</td>
<td>International Studies</td>
</tr>
<tr>
<td>Gamma</td>
<td>Student 3</td>
<td>Male</td>
<td>Theatre Arts</td>
</tr>
<tr>
<td>Gamma</td>
<td>Student 4</td>
<td>Female</td>
<td>Apprl, Housing, &amp; Resource Mgt</td>
</tr>
<tr>
<td>Gamma</td>
<td>Student 5</td>
<td>Male</td>
<td>Political Science</td>
</tr>
<tr>
<td>Gamma</td>
<td>Student 6</td>
<td>Male</td>
<td>Building Construction</td>
</tr>
</tbody>
</table>

4.4.4 Pilot Study

Prior to this dissertation study, a pilot study was conducted in fall of 2014 on the same CT course. The goal of the pilot study was to evaluate the key course design decisions and to assess if the main learning objectives of the course was being met. In the pilot study, both qualitative and quantitative data were collected. Qualitative data included 13 hours of group observations data and interviews with 9 students. Based on an observation check list (see Appendix A) it was found that students asked questions and provided explanations to each other, asked for technical assistance, discussed social life, and looked at each other’s code. Quantitative data included results from 20 students collected via a motivational survey, based on the MUSIC Model of Academic Motivation (a model of motivation broken down into five aspects: students’ perceptions of their empowerment, usefulness, success, interest, and caring) (Jones, 2009). One of the key findings of the pilot study is that students found the cohort (group) model a significant factor in motivation, with students citing it as helping their learning process while also being both interesting to their long-term goals (Kafura et al., 2015). In fact, students’ cohorts considered it almost as useful as the assistance from instructors. During interviews students also suggested that their group was a safe place to ask for help and receive explanations from peers. Overall, the pilot study highlighted various collaborative aspect of
learning as a salient feature of developing CT skills. However, from the pilot study it was not apparent how group interactions influenced learning of CT. Thus, this study is particularly focused on collaborative aspect of learning CT in a face-to-face classroom setting by building on the pilot study. By operationalizing Chi’s DOLA framework to categorize the social interactions of students in small groups, this study intends to illustrate the association between social interactions and learning of CT.

4.4.5 Data Collection

Prior to the start of the CT course in spring 2015, the researcher obtained Virginia Tech Human Subjects Research Approval through the IRB. Participant consent was obtained prior to the initiation of data collection procedures. The current IRB approval for this project is IRB 14-627. This study is part of the NSF-award 1444094 and NSF-award-1624320 investigating ‘Scaffolding Big Data for Authentic Learning of Computing’.

Data collection for this study was conducted in one academic term (semester system, approximately 15 weeks). The researchers regularly took field notes. Three student cohorts working during class time were video recorded during three class sessions. Each video was approximately 20 minutes in duration.

On the first and second day of the observation students were primarily working on the Blockly platform. Blockly is visual programming environment that uses blocks (see Figure 4.3) that link together to make writing code easier, and can generate JavaScript, Python, PHP or Dart code.

![Sample Blockly code](image)

Figure 4.3 Sample Blockly code

On the third day of observation students were working between Blockly and the Spyder (Python) editor, shifting between the two platforms (transferring from block based coding to text based coding). Figure 4.4 is the Blockly platform and Figure 4.5 is the Spyder platform.
The following section describes each goal of each observation day and the student assignments.

4.4.5.1 Observation day one
This was the 15th day of the CT class. The students have been re-introduced to the concept of abstraction and algorithms (e.g., decisions and iterations). On this day students were also introduced to the concept of dictionaries (a collection of key: value pairs). The class activity consisted of four questions requiring the students to:

- select items based on a certain criteria [ use/apply the concept of conditions]
• create a new list based on an old list [use/apply the concept of iteration, initializing a list, calculating a value based on an existing abstract value, and adding a calculated value to a list]

• create a new list based on an old list and certain criteria [use/apply the concept of iteration and condition, initializing a list, calculating a value based on an existing abstract value, and adding a calculated value to a list]

• extract certain part of data from a dictionary data structure [understand the concept of different types of data structure, dictionary in particular]

4.4.5.2 Observation day two
This was the 16th day of the CT class. The students have been familiarized to the concept of dictionary. On this day the expectation was that students will be feeling comfortable to manipulate the different parts of the dictionary. Students will be continuing to use decision and iteration on parts of the dictionary. The students are to do similar activities on completely new dictionary one that the instructor has not demonstrated on. The class activity consisted of five questions requiring students to:

• extract specific data from a dictionary [understand the structure of a dictionary, apply the concept of iteration]

• create a new list based on data extracted from the dictionary, plot them [understand the structure of a dictionary, apply the concept of iteration, create plots, initializing a list, adding an abstract value to the list, creating a calculation]

• do similar actions but on a different dictionary than the one students were initially familiarized with.

4.4.5.3 Observation day three
This was the 18th day of the CT class. At this point student have been introduced to text based programming in Python by showing students the equivalent text code of the visual code blocks they have been using. On this day students are asked to look into text code, find errors and fix errors of given text code, logical errors. Logic errors are errors that prevent a program from doing what you intended it to do. The code may compile and run without error, but the result of
an operation may produce a result that you did not expect. By fixing logical errors students are expected to get familiarized with text based programming language structure and syntax.

The class activity consisted of six questions that required students to:

- identify a logical error in two programs
- apply conditional logic and iteration in text based programming language. Most of the code students need to write for these four questions was provided in the first two programs.

According to the definitions of active, constructive and interactive behaviors provided in the DOLA framework all the class work problems were categorized as Constructive (see Table 4.4). This is because each day’s problems required the students to apply what was taught in class on new data sets, and had to gradually integrate different concepts.

Table 4.4 Type and concept covered in classwork problems during observations

<table>
<thead>
<tr>
<th>Observation Day</th>
<th>CT Concepts</th>
<th>Type of activity</th>
<th>No. of questions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Day 1</td>
<td>Conditional logic, Decision</td>
<td>Constructive</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>Iteration</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Dictionary</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Day 2</td>
<td>Iteration</td>
<td>Constructive</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>Dictionary</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Create plots</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Day 3</td>
<td>Identify logical error, Decision</td>
<td>Constructive</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>Conditional logic</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Iteration</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
4.4.6 Data Analysis

To analyze the interaction data the following eight steps suggested by M. T. Chi (1997) for verbal data analysis was followed.

1. Select the sample data set
2. Select the unit size of data for analysis
3. Choose or create a coding scheme
4. Choose how data coding will be implemented
5. Represent the coded data in a way that can be analyzed
6. Analyze the data
7. Interpret the analysis
8. Repeat necessary steps

First, the video recording of three days of observation was selected as the sample data set. These video recordings were transcribed and the transcribed data set was used as the sample data set. When necessary the coders would go back to video recordings to verify certain events if it was not clear from the transcripts.

Second, the unit for data analysis was considered to be episodes. Section 4.4.6.1 provides more detail on episodes.

Third, a coding scheme was developed from the pilot study based on Chi’s DOLA framework. This framework (see Table 4.5) was used to code data.

Fourth, each student in each episode was then characterized as either being active, constructive or interactive based on the adapted framework of Chi’s (2009) overt activity framework. Two independent coders coded all data. First the coders identified episodes in all transcripts. Second, the coders applied codes to each student participating in each episode based on their major role in that particular episode. In both cases, the coders compared their codes with each other. Disagreement between identifying episodes (18% disagreement) and applied codes (15% disagreement) were discussed between coders and resolved (Creswell, 2008).

Fifth, in order depict the qualitative data, the number of times a student exhibited a certain behavior was tallied. Tables presenting the frequency of different overt behaviors for
each cohort was also created. This allowed the researcher to gain an overall interpretation of cohort member behavior and to detect patterns across cohorts. Quantifying qualitative data in such a manner has been described by Chi (1997) as process where the “researchers rely strictly on the qualitative data, but they quantify the analyses” (p.7). This allowed the researcher to compare and confirm her subjective interpretation from transcribed data with frequencies of the codes quantitatively. The following section provides the definition of episodes that has been used in this study.

4.4.6.1 Episodes

In order to characterize student’s overt behaviors daily transcribed dialogues were segmented into episodes. “Episodes are characterized as coherent sequences of sentences of a discourse, linguistically marked for beginning and/or end, and further defined in terms of some kind of ‘thematic unity’--for instance, in terms of identical participants, time, location or global event or action. (Van Dijk, 1982, p.2). In this study episodes were identified when students within a group were saying or physically doing something with each other or talking out loud to him/herself. The starting of an episode was marked when a student within a group asked for help or said something out loud, expressed some sort of concern. The ending of an episode was marked when someone responded to the member’s inquiry or when the questioner him/herself showed some sort of acknowledgment, or if the observer noticed that none of the group members had responded to the questioner. Following is an example of an episode.

*Student A:*  
* Student A:  
Huh, okay, I might be getting it more. Why did I keep this? This is the ‘feels like temperature’ is equal to ‘this’, right? It’s finding the temperature, like the ‘feels like temperatures’, because that’s what we’re trying to find?

*Student B:*  
That's what I did ‘temperature’.

Each student in each episode was assigned a code based on the primary role the student exhibited in that episode. For example, in the above excerpt student A was assigned a constructive code and student B was assigned an active code.
4.5 Findings

The findings of this study are organized into the following sections:

- Section 4.5.1 describes the adapted framework for evaluating active, constructive and interactive in the context of learning CT
- Section 4.5.2 presents descriptions of each cohort and it’s observable characteristics
- Section 4.5.3 describes the cross-cohort comparison

4.5.1 Chi’s Adapted Framework in the context of Learning CT

In analyzing the overt activities of cohort members, the framework developed by Chi et al. (2009) was used as a starting point. Table 4.5 provides the primary and emerged codes, definitions and examples of quotes used to code the episodes analyzed during observations. Since no episode was considered as being “interactive” the adapted framework does not include example quotes in this category.

Table 4.5 Operationalized overt behaviors in the context of learning CT based on Chi (2009)

<table>
<thead>
<tr>
<th>Type of overt activity</th>
<th>Code and definition</th>
<th>Example quote</th>
</tr>
</thead>
<tbody>
<tr>
<td>Active</td>
<td>Look/Show: A student shows another student his/her own code. No elaborate explanation is provided.</td>
<td>A student is looking over another student’s computer.</td>
</tr>
<tr>
<td></td>
<td><strong>Design Major:</strong> Did you get no. 3? <strong>Public Relations:</strong> Yeah. <strong>Design Major:</strong> Did you basically just do like number line...(shows the computer screen) <strong>Public Relations:</strong> I will show it to you.</td>
<td></td>
</tr>
<tr>
<td>Active</td>
<td>Ask a shallow question: A student asks another student/TA to verify, complete a problem.</td>
<td><strong>International Studies:</strong> For number 4. we disregard wind, right?</td>
</tr>
<tr>
<td>Active</td>
<td>Self-talk: When a student is trying to solve a problem and provides a live commentary of what s/he is doing/ says out loud what s/he expects the code to do</td>
<td><strong>International Studies:</strong> You got it. Output. Print that thing. No. Print current temp duh. You've got this. Don't give up Oh, you! I said you can do it but you obviously can’t.</td>
</tr>
<tr>
<td>Struggle</td>
<td>Struggling:</td>
<td><strong>Political Science:</strong> So we would have to ... There would have to be a ...</td>
</tr>
</tbody>
</table>
Both students are trying to solve a problem independently but discuss issues with each other, both students do not give any concrete feedback.

| Constructive | Explain to other: In response to another student’s question a student explains in his/her own words how to solve a problem. This can include the student showing his own code. | Political Science: You have to have a set. Political Science: I don’t know. So did you have to have like two sets of property blocks? Theater Arts: I have no idea what I’m doing. Political Science: don’t know what to do. I’m happy that the work today is not doing these blocks. I’m actually really happy. Theater Arts: What is this? |
| Constructive | Self-monitor/Reflect: A student expresses what s/he thinks s/he might know or does not know | Political Science: Like you want me to like put this into here? Science Major: This, I mean, because ... You're checking to see if the depth is less than six. And then so you're saying it is, so we now need to change it into miles, and then add it to a list. So instead of printing, you would have that collision...and then change height to depth. So to List, New List, then you just, you come out. New list needs to be like three down, so take, opens it up yeah, disconnect. With the new list, set clock, click it one more time. Then, go back into your list the list. You should be able to do it now. You're checking to see if it's six. |
| Constructive | Ask an elaborate question: A student asking the question rephrases the question stated in the problem or involves conceptual information or asks a question relating to the process of solving the problem. | Political Science: Huh, okay. I might be getting it more. Why did I keep this? This is the feel like temperature equal to this, right? It’s finding the temperature, like feels like temperatures, because that’s what we’re trying to find? |
| Constructive | Plan/Connect A student connects or links a problem to another similar problem. Can plan how to solve the problem. | Political Science: Okay. Isn’t three essentially the same thing as two then? Theater Arts: Yeah, except that you're only going to print earthquakes that are shallow in miles. You have to convert ... You have to find ones that are less than six kilometers and then convert that. |
Interactive | Build on partner’s contribution (argue, defend, confront or challenge) Each student is roughly equally participating/contributing in a conversation among students. | Did not find any

Interactive | Revise errors from corrective feedback A student receives feedback from another person and is able to fix his/her misconception or solve the problem. If there is a conversation between the expert and novice, where each participates more or less equally then the episode is considered to be interactive. | Did not find any

In the context of students learning CT in a classroom setting, several themes suggested by Chi’s overt activities framework emerged naturally. In terms of being active, the category show/look was applied to number of episodes. In terms of being constructive, the category explain to other self-monitor, connect/plan were observed.

However, several categories emerged that were not explicitly stated in Chi’s (2009) DOLA framework. These are: asking a shallow question vs asking an elaborate question, socializing, being confused or expressing frustration, self-talk and struggle. The following paragraphs further describe and justify each category of codes.

4.5.1.1 Asking a Shallow vs. Elaborate Question

Chi (2009) suggests that “asking a question” to be considered as a constructive activity. Being constructive requires a learner to state something that is not the verbatim of what is given to the learner. It suggests that the learner is integrating and making inferences about the concept that is not explicitly given to him/her (Chi, 2009, 2015). While coding questions the researcher found that not all questions students asked were constructive in nature. At times students asked questions in which they were going beyond what had been given to them in the study material. However, most of the questions were not of that type. Instead, students simply repeated the
questions stated in the classwork problem, or asked for direct help without any elaboration of the concept or the problem being solved. This type of asking question suggested more of an active behavior.

Thus, asking questions was divided into two categories: asking an elaborate question and asking a shallow question. Asking an elaborate question was considered to be constructive and asking a shallow question was considered to be active. In the following quote an International Studies student was asking a shallow question.

*International Sciences Student:* For number 4 we disregard wind right? *(asking a shallow question)*

In the following quote a student in Architecture was asking an elaborate question.

*Architecture student:* Did you have your iteration and then your decision inside the iteration? *(asking an elaborate question)*

### 4.5.1.2 Self-Talk

While a student solved problems, s/he gave a live commentary of what s/he was doing. It seemed like the student was talking with the computer. This type of activity was coded as self-talk and was considered an active behavior. The justification for self-talk to be active was that the student was speaking out loud what s/he was doing or trying to do. Since, showing or looking at a computer was considered to be active, self-talk was also considered to be active. In the following two separate quotes we see an English major and an International Studies talking to themselves while coding on their own computers.

*English major:* *(while working on the computer)* Oh shoot I forgot to add...

*International Studies:* *(while working on the computer)*

  You got it. Output. Print that thing. No. Print current temp duh.
  You've got this. Don't give up
  Oh, you! I said you can do it but you obviously can't.

### 4.5.1.3 Struggling

In certain episodes, one or more students were exhibiting sequence of back and forth statements. In such episodes neither student gave a direct definite response. One student would be trying
different options on his/her computer program, and exploring half-formed ideas (which were at times also incorrect). Such episodes were coded as struggling. Since it was difficult to determine if the student was being active or constructive, it was coded as a separate category altogether. This struggling state was often noticeable among the novice students, especially among the student with non-science background. This could be due to the fact that the syntax and semantics of writing code have to be exact in order for a problem to be solved correctly. If either student in a discussion did not understand the computational concept and ways to implement it in the code, the students were found to be in a struggling state. In the following excerpt we see two students struggling to extract values from a dictionary in Python text editor.

Housing student: Maybe it’s not report.

International Studies Student: It’s not report. I just changed it to that.

Housing student: See, it’s reporting, reports.

International Studies Student: No that doesn’t work either. I’ve already done that.

Housing student: Maybe it’s uh .. It should be, it’s ... uh ... run...

In the above excerpt the two students were going back and forth and trying out different options since neither of them had a clear idea how to solve the problem. For a student to overcome a struggling episode the students actively or constructively reached out to others in the cohort or ask help from the TA. Figure 4.6 illustrates the adapted coding framework characterizing students’ naturally occurring overt CT learning behaviour.
Figure 4.6 Adapted framework for categorizing naturally occurring CT overt behaviors

4.5.2 Observable characteristics of each Cohort

This section describes the findings of each individual cohort based on the analysis using the adapted CT framework.

4.5.2.1 The Alpha Cohort

The Alpha cohort had five students, three females and two males. However, during the days of observation one team member (female) was not present in all three days. Most of the students were enrolled in a liberal arts disciplines, one student had a design background.
Evidence from the observation showed that students of this cohort were more active than constructive. Figure 4.7 Percentage count of different overt behaviors exhibited by the Alpha cohort shows that 69% of student’s overt behaviors of the Alpha cohort were active in nature, 27% were constructive and 4% were of the nature of struggle.

![Alpha Cohort](image)

At the individual level, it was observed that the student from International studies (21 episodes) exhibited the most active overt behavior compared to the others in the cohort (Theater arts student: 9 episodes, Architecture student: 12 episodes, Political science student: 16 episodes). In terms of being constructive, students from Theater arts (7 episodes) and Architecture (6 episodes) exhibited some constructive overt behavior in the cohort. In terms of struggling, student of theater arts (3 episodes) and student from political science (2 episodes) struggled the most. Figure 4.8 comparatively illustrates the number of different types of overt behaviors exhibited by each student of the Alpha cohort. In general, all four students that were present during the observations exhibited active, constructive and struggling behavior.
Figure 4.8 Episode count of different overt behaviors exhibited by each student of the Alpha cohort
Further sub categorization of active behaviors (see Figure 4.9) reveal that students in the Alpha cohort mostly asked shallow questions to each other, showed each other how to solve a problem, talked to themselves and socialized with each other. Sub categorization of constructive behaviors reveal that students of the Alpha cohort at times did ask elaborate questions and explain each other a problem. Also while explaining a problem the Architecture Student and Theater arts student planed and connected their solution with previous problems. The TA of the Alpha cohort mostly explained the solution to the students or showed the student how to solve the problem.

![Alpha Cohort Behaviors](image)

Figure 4.9 Percentage count of different types of active, constructive and struggling behaviors exhibited by each student of the Alpha cohort

In summary, the interactions between the members of the Alpha cohort was mostly active in nature. Collaboration was mainly between dyads. Students in this cohort mostly asked shallow questions to each other and instead of explaining the problem, usually showed each other what they had done. Among the four students who were usually present in class, Student 1 and student
4 exhibited more constructive behavior than the other two students. The students socialized with each other and when necessary asked help from the TA.

4.5.2.2 The Beta Cohort

The Beta cohort comprised three female students and three male students. Four students this cohort were enrolled in the liberal arts discipline, one student in science discipline and another in a design discipline. All students were present during days of observation. One student of the Beta cohort had previously taken courses in programming. For the rest, this CT course was the first computer science/programming course.

The observation data analyzed using the adapted overt activity framework revealed (see Figure 4.10) that the Beta’s overt behaviors were mostly active (56% of episodes) in nature. 41% of all activity of the cohort was constructive and only 3% of episodes were struggling episodes.

![Figure 4.10 Percentage count of different overt behaviors exhibited by the Beta cohort](image)

At the individual level, we see student 5 to be a student that exhibited most overt behaviors (29 episodes). He was the one who was most constructive (21 episodes) in the cohort. It is worth mentioning that student 5 had prior programming experience. Other students who exhibited constructive overt behavior were student 1 (5 episodes) and student 3 (5 episodes). Student 1 exhibited the most active overt behavior (15 episodes) amongst the Beta. Student 2, 4 and 6 exhibited more or less the same amount of active (4-8 episodes each) and struggling (1 episodes...
each) overt behavior. The TA of the Beta cohort constructively (4 episodes) helped the students of the cohort. Figure 4.11 illustrates episode count of different overt behaviors exhibited by each student of the Beta cohort.

![Beta Cohort Diagram]

Figure 4.11 Episode count of different overt behaviors exhibited by each student of the Beta cohort

Further sub categorization of active and constructive behaviors reveals that the student from Chemistry of the Beta cohort was responsible for most of the explaining in the cohort, planned or connected problems with previous problems, or showed another student how to solve a problem. The chemistry major also socialized with others in the cohort. The other students mostly asked shallow questions, show or looked at each other’s problems, asked elaborate questions and socialized. The English and Political Science major also exhibited self-reflective behavior. Figure 4.12 shows percentage count of different types of active, constructive and struggling behaviors exhibited by each The TA of the Beta cohort explained the problems to the students.
Figure 4.12 Percentage count of different types of active, constructive and struggling behaviors exhibited by each student of the Beta cohort

4.5.2.3 The Gamma Cohort

The Gamma cohort had three female students and three male students. Four students had a liberal arts background, one had a management background and the other student had a science background. All six students were present during the first two days of observation. Student 5 and 6 were absent on the third day. Apart from student 6, the other five students had taken the CT course as their first computer science/programming course.

Evidence from the observation showed that students of this cohort were more active than constructive. Figure 4.13 shows that 61% of the overt behaviors of the Gamma cohort were active in nature, 9% were constructive and 30% were the nature of struggle.
At the individual level (see Figure 4.14), it was also observed that students exhibited active overt behavior compared to struggling behavior. The student majoring in International studies exhibited some constructive behavior. Students majoring in Public Relations, International Studies, Political Science and Appeal, Housing and Resource Management were mostly seen to struggle in the Gamma cohort.
Figure 4.14 Episode count of different overt behaviors exhibited by each student of the Gamma cohort

Further subcategorization (see Figure 4.15) of active behavior revealed that students’ active behavior consists of asking shallow questions and showing or looking at another student’s code. Students of the Gamma cohort were not seen to socialize or self-talk. Sub categorization of constructive behaviors reveals that students of the Gamma cohort ask shallow questions, show each other how to solve a problem and struggled. In terms of constructive behavior only the student majoring International Studies asked elaborate questions. The TA of the Gamma cohort explained the solution of problems to the students.
4.5.3 Cross-Cohort Comparison

This section reports the findings from the cross-cohort analysis. Common characteristics across cohorts as well as unique characteristics of the cohorts are described based on the adapted CT overt framework (section 4.5.1).

Figure 4.15 compares the behaviors of all three cohorts. Active overt behavior were observed more than constructive and struggle. The Alpha cohort exhibited more active overt behavior (59 episodes) than the Beta cohort (48 episodes) and the Gamma cohort (33 episodes). Members of the Beta cohort exhibited more constructive overt behavior (31 episodes) than the other two cohorts (Alpha: 21 episodes, Gamma: 5 episodes). In terms of struggling, the Gamma cohort struggled more (16 episodes) than the Alpha cohort (3 episodes) and the Beta cohort (3 episodes). Both for the Alpha and the Beta cohorts, active overt behaviors were exhibited the
most, the next was constructive behavior. In the case of the Gamma cohort, struggling episodes (16) were observed more than constructive episodes (5). No interactive behaviors were observed in any cohort. Figure 4.16 summarizes overt behaviors exhibited by each cohort.

![Summary of different types of activities exhibited by cohorts](chart)

Figure 4.16 Number of episodes of overt behavior exhibited by each cohort during observation
Figure 4.17 illustrates cross-cohort comparison of subcategories of overt behavior exhibited by students enrolled in a major in the College of Liberal Arts (8 students), College of Architecture (4 students) and College of Science (1 student). In terms of active behavior students of three colleges asked a shallow question, showed or looked at another student’s code, self-talked and socialized. In constructive behavior, students asked elaborate questions, explained in detail how to solve a problem, self-reflects and planned out their solution process. Self-reflection was observed in the English major and a Political Science major only.
Table 4.6 Summary of overt activity episodes of students coded as Active, Constructive, Interactive and Struggling across cohorts

<table>
<thead>
<tr>
<th></th>
<th>Cohort</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Alpha</td>
</tr>
<tr>
<td><strong>Active</strong></td>
<td></td>
</tr>
<tr>
<td>Ask a shallow question</td>
<td>24</td>
</tr>
<tr>
<td>Show or look at another student’s solved problem</td>
<td>20</td>
</tr>
<tr>
<td>Self-talk</td>
<td>3</td>
</tr>
<tr>
<td>Socializing</td>
<td>12</td>
</tr>
<tr>
<td><strong>Constructive</strong></td>
<td></td>
</tr>
<tr>
<td>Ask an elaborate question</td>
<td>6</td>
</tr>
<tr>
<td>Explain other</td>
<td>12</td>
</tr>
<tr>
<td>Plan/Connect</td>
<td>3</td>
</tr>
<tr>
<td>Self-reflect</td>
<td>0</td>
</tr>
<tr>
<td><strong>Interactive</strong></td>
<td></td>
</tr>
<tr>
<td>Build on partner’s contribution</td>
<td>0</td>
</tr>
<tr>
<td>Revise error</td>
<td>0</td>
</tr>
<tr>
<td><strong>Struggling</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>9</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>89</strong></td>
</tr>
</tbody>
</table>

The adapted framework for characterizing students’ collaborative overt behavior while learning CT allowed us to better understand how students interacted with each other and how it impacted their learning. The findings of this study indicate students’ overt CT learning behaviors to be mostly active in nature. Constructive behavior was present but was limited in number. Pair of students struggled while learning CT. No interactive episodes were identified in this study. The following sections elaborate these findings and suggest why certain trends of behavior were noticed in this study.
Of all observed behaviors, 87% was accounted for by just five behaviors: asking a shallow question, show/look at another student’s solved problem, ask an elaborate question explain to other and struggling on a task (illustrated in Figure 4.18).

4.6 Discussion

Findings of this study suggest that students in this particular CT class were mostly active and struggling in their interactions. Some constructive interactions were noticed. No interactive interactions were observed. The following sections elaborate on these findings.

4.6.1 Why Were Students ‘Active’ In Their Overt Behavior In The Ct Class?

The findings of this study clearly suggest that this particular group of students (mostly majoring in non-science and non-engineering disciplines) were active in nature while communicating with each other. For most of the students, this CT class was the first computer science/programming related course they had taken. The terminology was new, and the way of thinking was new too. While interacting with each other, instead of explicitly saying what the problem was, students would instead show their computer screen, look at another’s solved solution – all of which were
active in nature, but which indicated a lack of terminology and cognitive ability to frame a verbal description. The following section provides few active CT episodes exhibited by students.

4.6.1.1 Getting started: Active
Getting started with a problem was common active behavior students exhibited in the CT class. In multiple episodes, a student was seen to ask another cohort member or the TA what should be the first step for solving a problem. In following excerpt we see a student from Architecture asking help to get started.

Architecture student: I kind of missed...How do we start this one?

In another episode, even though the TA had explained the whole process of how to extract certain values from an existing list, after the discussion, the Theater arts student asked the TA

Theater arts student: Okay, so I should ...What should be the first step?

For novice CT learners of this study, getting started to write code was a common hurdle.

4.6.1.2 Deciding on variable names: Active
Declaring a variable and deciding on what a variable’s name should be was another common active problem students faced. Variables are used to store information to be referenced and manipulated in a computer program. They also provide a way of labeling data with a descriptive name, so that programs can be understood more clearly by the reader. It is helpful to think of variables as containers that hold information. Their sole purpose is to label and store data in memory. The concept of a variable as the name for a (perhaps mutable) value of a given type is a fundamental concept that novice learners of CT need to understand.

One of the class work assignments required the students to create a new variable and the name of the variable was not specified in the question. This was a good exercise for the students since having variable name explicitly specified in questions may lead students to think that a variable has to be of a particular name. For example, an International Studies student was asking a student from Architecture what variable name he had used.

International Studies Student: What did you call that list? Just curious.

Architecture Student: I just called it “list in miles”. Yeah, list in miles...
Similarly, we see another episode a student another whether the variable name was correct.

Political Science Student: First you just set your property name?

Science Student: I called it Weather Dictionary. The names are just so you know, so you can get the values by referring to that name

In both of the above excerpts we see students verifying variables names given to new lists.

4.6.1.3 How to decide when to use a certain computational block: Active

Students seemed to be confused about different blocks and when to use which one. In multiple episodes students asked each another “which block to use”. Also, instead of calling blocks by their computational terminology, students would ask the color of the block (in this case Blockly block colors). This could be because the computational terms were new to the students and some of them still haven’t quite grasped the understanding of the meaning of each block. Furthermore, since a problem can be solved in multiple ways, this made it even more difficult for novice CT learners to understand when to use which block or which approach to solve a problem. In the following excerpt we see a student from Theater arts interacting with a student from International studies.

Theater arts student: I find this weird. How do you know when to use certain blocks? I'm confused by that. Is it trial by error?

International studies student: It is a lot of times. When it comes to this stuff it's all, it's actually really helpful to do that. If it's wrong, you kill it and keep going at it.

Most programming problems can be solved in multiple ways. In the above excerpt both students are in an active mode. As a novice learner the International studies student expresses his limited understanding of when and why certain blocks are used. However the response suggested more of a trial and error approach instead of elaborating on different approaches one can take.

In a different episode we see students referring to the color of a programming block in Blockly instead of using the computational term.

Housing Student: I think this is when we use the thing from the other day.
International Studies Student: The red thing? Do you just print the values?

Housing: Print, yeah.

International Studies Student: I got it, I got it.

Weintrop and Wilensky (2015a) in their study on block based programming environment vs. text based programming environment also found that the visual cues: color and shape makes it easier for students to learn programming. The study also refers to blocks as memory aids where the environments provides an easy and organized way to browse all the available blocks (p.204). In the above excerpt we also notice the international Studies student referring to a computational block by its color.

4.6.2 When and why were students constructive?

Some cohorts exhibited more constructive behavior than the others. Prior experience in programming and working on a problem for a while allowed students to be constructive.

The science major in the Gamma cohort who exhibited more constructive behavior than others had prior programing experience. He was also communicative and concerned about other learners of the cohort and tried to help his peers. Not only the science major in the Gamma cohort but also in the design major in the Alpha cohort who socialized and was actively involved in dialogue with other students exhibited constructive behavior. When a student is invested in helping other students learn, s/he could be seen to be constructive, trying to elaborate the concept or problem being solved and helping the peer student gain a better understanding of the concept (See Figure 18). The lack of socialization episodes could be another reason for why there was less interactions in the Gamma cohort.

Another situation where students were constructive was when they asked a question to the TA after struggling on a problem. In most episodes students who were struggling to solve a problem would try multiple things, ask for help from their peer cohort members and only then ask for help from the TA. Prior to asking help from a TA a student had worked on the problem for a while, and had solved certain parts of the problem. As a result while asking a question to the TA the student would explain what s/he had done or tried and elaborate on his/her own understanding,
exhibiting a constructive behavior. In following excerpt we see a Theater Arts student asking an elaborate question to the TA.

Theater Arts Student: If the earthquake is less than 6 and it gets moved to the list of shallow earthquakes and then it just like (shows computer screen)

TA: You've made it a little bit more complicated than it needs to be. You have the right idea. So, what you did here, all you need to do is you need to look for those that are less than 6 then you can convert those to miles right away

While being constructive, the student elaborated on computational concepts and terminology. The following sections describe these constructive CT episodes.

4.6.2.1 Trying to better understand the meaning of computational terminology:

Constructive

Computational terminology e.g. append, iterate, condition can be confusing for novice learners. In the following excerpt a student from political science was struggling with the term “append” (e.g., the Python operation to add an element to a list). A student from Theater Arts expressed the idea of moving and appending to be similar.

Political Science Student: What does "append” block mean?

Theater Arts Student: Move. You're taking that and putting it on this list.

Political Science Student: Append essentially just means "move around?"

Theater Arts Student: Yeah, like, put that on the list.

We see both students elaborating on the terminology of ‘append’ in the above excerpt. Although the explanation of moving might suggests that the Theater Arts student thinks the original value is destroyed (which would be a wrong assumption) as opposed to being copied, both students are trying to better understand how a the certain block operates and its name associated to it.

4.6.2.2 Providing analogies to better understand a concept: Constructive

Analogies sometimes help novice learners to better understand computational concepts. The following excerpt is an example of students using common analogies to better understand computational concepts, in this particular case a list of dictionaries.
Science Major: This is the list of each of those dictionaries. So when it says, “this is the current item”. Like this dictionary.

English Major: Is each thing in this dictionary, a dictionary of dictionaries?

Science Major: Mm-hmm (affirmative).

English Major: The dictionary has words and values.

Science Major: The word’s the key, remember? You can say dictionary of the temperature...

English Major: Unlocks the door to the value?

Science Major: Yes, and it says 30. The humidity is a value of 20. So wind speed you know. You say you need 30. Instead of having to iterate through them you go in and say I need this out of this out of this thing. You just get it and pull it right out.

English Major: The key that opens the door to the integer.

Science Major: There you go. If you want to think about it like that.

In the above episode a Science major was explaining to an English major what a list of dictionaries was and its structure. In the course of the discussion, the English major came up with her own analogy of why the term “key” was used in the case of a dictionary: “the key that opens the door to the integer”. Using analogies to remediate misconceptions and foster conceptual understanding has been suggested by D. E. Brown (1992) in studying concepts in physics. Computational concepts can also be taught through analogies. In the above excerpt we see peer students taking this analogy approach of understanding a computational concept. However, only coming up with analogies or examples is not sufficient to form a conceptual understanding. A student will also need to understand why the concept should be applied to particular situations (D. E. Brown, 1992).

4.6.2.3 The concepts of list and dictionaries: Constructive

Manipulating lists and dictionaries was a common problem students needed help with. Since the concept of list was new to novice CT learners, the person assisting the students needed to explain
how to solve the problem. In following excerpt the Science major was explaining how to manipulate a list of dictionaries to a Political science student.

*Political Science student:* How do I print the dictionary? It’s- [showing what she had done on her computer screen]

*Science student:* Okay.

*Political Science student:* These 3 are dictionaries, right?

*Science student:* Yes.

*Political Science student:* [looking at the computer screen of the Science student] Yeah, well, I had that. It told me I was wrong. I had it in the exact same format as it was in.

*Science student:* list of dictionaries, so for in Arizona forecast. You have to print out

*Political Science student:* Am I printing out words or am I printing out temperatures?

*Science student:* You print out each of them.

*Political Science student:* Each what?

*Science Student:* Each temperature in the dictionary. Just in the dictionary.

*Political Science student:* But the words, or the values?

*Science student:* This is a list of dictionaries, so it's going to give you one dictionary, and then the next dictionary back, and then the next dictionary back. Click that and we’ll get rid of the other.

*Political Science student:* So I’m printing out-

*Science student:* I just named it dictionary.

*Political Science student:* Each dictionary. Print the dictionary?

*Science student:* It’ll give the .Yeah, because it works; that’s exactly what works for mine. “You got it! Now just print out the next bit.”

In the above excerpts the student majoring in science had to elaborately explain how a value of a list is extracted and what that value would look like.
4.6.3 Why did the students struggle instead of being active or constructive?

Students in the CT class struggled to solve a problem when they did not have a clear idea of the concept or did not know how to perform a certain task in the software. This occurred more often when peers were at the same level of understanding. If any of the students in the group could not figure out how to solve the problem, they kept on trying different things, hoping “by chance” things would start working. Often after a while one of struggling students would ask help from the TA or someone in their group.

When students did not know how to solve a problem and something new was being taught for the first time, they struggled the most. The following section explains such struggling CT episodes.

4.6.3.1 Both students did not know how to solve a problem: Struggling

Before the students started to solve their daily class work problem, the instructor of the course gave a short lecture and demonstrated programming techniques related to the concepts being taught that day. Presentation slides and reading materials covering the concept were available online. A hyper link to these materials was provided in the computer interface right above where the students worked on their class work problems. This was done intentionally by course designers so that the students could have easy access to the material that covers the problems that needed to be solved on a particular day. However, students rarely went back to read a topic. This was an example of students not being self-reflective (in such case the student would realize s/he needs to re-read the study material or look at examples) instead they kept on struggling.

*International Studies Student:* Something is missing here. It is trying the right number. But it’s not printing its state

*Housing Student:* Wait, why is this (pointing at peer’s computer screen)?

*International Studies Student:* Yeah that's what I'm thinking. I think that's not what it's looking for.

*Housing Student:* Print the lowest?

*International Studies Student:* Yeah, but we also, like know the lowest state.

*Housing Student:* Lowest. Oh safest! Whatever you set it for!
International Studies Student: The lowest rate just prints out a zero so it’s calculating something wrong.

Housing Student: Should it be printing the safest state? Anything you print in parentheses, its printing for me.

International Studies Student: I ... yeah, me too.

In the above excerpt we see both students were struggling to fix a piece of Python code. The program the students were working on was supposed to generate the “safest state to live in” and “the state that has the lowest violent crime rate”. They were given a dictionary that contained a state’s name and crime rate. Writing text based programming code was new to these students. They had used block based programming language in the previous weeks. So they were familiar with the terminology and structure of a dictionary but we saw them struggling to figure out how to extract a particular value from a dictionary in a Python text editor. Since none of the students have figured out how to do it, they are struggling. It may be noted that the students could have looked at example codes provided by the instructor at the beginning of the class where similar problems had been described.

4.6.4 Why the absence of interactive overt behavior?

Being interactive requires multiple students to contribute equally in a conversation and build on each other’s ideas (Chi, 2009). Studies (Michelene Chi & Menekse, 2015; Michelene Chi & Wylie, 2014; Menekse et al., 2013) that examined interactive behavior had students solving a single problem together and had to come up with a single solution. In such situations, students were more likely to debate each other and exhibit interactive behavior.

In this particular study, students did not exhibit interactive overt behavior. This means students did not equally contribute in a discussion and elaborate/debate on another’s understanding on particular topic. There are few reasons as to why only one sided conversations were noticed. First, students were solving class work problems independently instead of collaboratively solving a single problem. Second, these class work problems were constructive by design, meaning, the classwork problem required students to apply and combine multiple concepts and skills learned on different data sets. Third, there was no direction that the students must dialogue with each other. If a student needed help he would spontaneously ask for help from another student or TA. Conversations in this particular group of novice learners of CT were limited to, mostly, pointing
and showing code to each other instead of verbally explaining to one another. Even if there was an elaborate conversation, it was mostly one sided. Thus with limited interactive exchanges, overt behavior was not noticed in this study.

4.6.5 What was the influence of student overt behavior on individual learning outcome?

This section examines observed behaviors, class work problems solved and grades received by each cohort member to gain an overall understanding of the influence (if any) of cohort interactions with learning. Number of class work problems solved was considered as an immediate learning outcome and course grade as distal learning outcome. Figure 4.19Figure 4.21 illustrate total number of different types of overt behaviors a student was involved in, total number of classwork problems solved during the three observation days and the final grade received by students of cohorts Alpha, Beta and Gamma.

![Alpha Cohort](image)

Figure 4.19 Total number struggling, active and constructive overt behaviors a student was involved in, total number of classwork problems solved during the three observation days and the final grade received by students of cohort Alpha

90
Figure 4.20 Total number struggling, active and constructive overt behaviors a student was involved in, total number of classwork problems solved during the three observation days and the final grade received by students of cohort Beta.
Each cohort had one or more distinct types of students. The Alpha had a student who rarely came to the class but completed the final project (which was worth 70% of the course grade). The Beta cohort had a student who attended all classes but finally withdrew from the course, and another student with prior programming experience who was willing to help out everyone in the cohort. The Gamma cohort had a student who did not interact with others but got an A; another student withdrew from the class.

The Alpha and Beta cohort seemed to be a collaboratively dynamic cohorts. From Table 4.7 we can see that on average each student of the Alpha cohort interacted with each other in 22 episodes, number of average class work problems completed was 3. On average each student of the Beta cohort interacted with each other in 14 episodes, number of average class work problems completed was 4. In terms grades received, as a cohort the Alpha scored similar to the Beta cohort (average course grade of cohort the Alpha was 3.2, the Beta’s average grade was
3.3). However, members of the Beta cohort on average exhibited more constructive overt behaviors (7 episodes) than the other two cohorts (Alpha: 5 episodes, Gamma: 1 episode).

Table 4.7 Average cohort interactions, classwork problems solved and grade received

<table>
<thead>
<tr>
<th>Cohort</th>
<th>Number of student present during observations</th>
<th>Average interactions</th>
<th>Average Active interaction</th>
<th>Average Constructive interaction</th>
<th>Average Class work problems solved</th>
<th>Average grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alpha</td>
<td>4</td>
<td>22</td>
<td>14</td>
<td>5</td>
<td>3</td>
<td>3.2</td>
</tr>
<tr>
<td>Beta</td>
<td>6</td>
<td>14</td>
<td>8</td>
<td>7</td>
<td>4</td>
<td>3.3</td>
</tr>
<tr>
<td>Gamma</td>
<td>6/5</td>
<td>13</td>
<td>5.5</td>
<td>1</td>
<td>5</td>
<td>2.7</td>
</tr>
</tbody>
</table>

The Gamma cohort, however, as a cohort did not seem to be that collaboratively effective in class. The average number of classwork by Gamma cohort members was the highest (5 classwork problems compared to 3 of Alpha and 4 of Beta cohort). However, in terms of interactions the Gamma cohort had the lowest number of interactions (average interactions 13) and the most struggling (number of struggling interaction 16) episodes amongst the three cohorts. During observations this low interactions was obviously noticed between Gamma cohort members. One student of the Gamma cohort had a head phone on his head most of the times, and another student would be working on his own. Mainly two students would be helping each other out.

In summary, no significant relationship of type of interactions between cohort members/ cohort behavior and immediate and distal learning outcome could be made.

4.6.6 Implications for Practice

Social interaction, which is key to any collaborative effort, was recorded, examined, and evaluated to measure student engagement in learning CT. The outcome of the study with the adapted DOLA framework can be used as an assessment tool to evaluate individual student’s collaborative learning of CT.

The adapted framework also illustrates how students may interact with each other in a cohort model. The framework can guide educators to better understand the kind of interactions to
anticipate. For example, the study suggests that novice learners would struggle or be active in nature. Thus, a computer science educator may consider providing additional active or constructive support so that learners can make transition from a struggling/active state to a constructive state. Also, an educator can keep an eye on students that are struggling. In forming cohorts it is beneficial to distribute learners with prior programming experience to different cohorts so that each cohort has at least one expert to serve as a means to help others in the cohort who are struggling.

The cohort model was found to support the active learning that the course was designed to achieve. This is seen in the active and constructive modes of student behavior. The findings of the study also suggest that it is important for all cohort members to regularly attend classes. Thus, class attendance should be given priority. Also, apart from learning related behaviors, informal, purely “social” discussions were found to be important because it established a setting in which students felt comfortable to ask help from cohort members. Thus, educators may want to create activities at the beginning of the semester that directly or indirectly encourage cohort members to become familiar with each other.

The findings of study also provide educators of CT with the concepts that novice learners find difficulty with. Breaking a problem into computationally solvable parts, naming and assigning names to variables are few basic tasks novice learners struggle with most. Although the students of this study did not solve problems together, still having a small group of students working and learning together was found to be beneficial. Even though not all interactions between members were learning oriented, having a group of learners encouraged to support each other created a positive learning environment.

The observed behavior of all of the student's in a cohort getting "stuck" implied that outside expertise must be fairly rapid to avoid too much dead time and/or frustration. This means that the ratio of "staff" (instructor, GTA, UTA) must be fairly small. Thus, this cohort model has difficulty scaling unless there are enough students with prior programming experience to serve as the expertise to get the cohort "unstuck".

If all of the students have no prior programming experience (almost always the case in the CT class and likely other settings) then the only way to scale the class (assuming staff is not
abundantly available) is through technology that provides rapid and "intelligent" feedback. However, the integration of such support may reduce amount of social interactions between students.

4.6.7 Limitations and Future Directions

This study only focused on the nature and quality of interactions between novice CT learners. No causal relationship was studied. For example, the role of disciplinary background or gender was not examined. Also, the academic level of a student was not considered during analysis. Future research could consider disciplinary major and academic level’s role in learning CT. Also, relationship between rate and quality overt behavior affecting individual or cohort learning outcomes can further be studied.

4.7 Conclusion

In a class teaching introductory CT to non-CS-majors, forming long term collaborative groups/cohorts was a beneficial teaching strategy. The cohorts allowed students to receive first line of support, creating a community where students from different disciplines could share their understanding of the new concepts and tools and as well voice their inabilities. Novice learners of CT in this class while solving a particular problem transitioned through stages such as struggling, being active and being constructive (see Figure 4.22). The transition process and its outcomes were influenced by the type of support and feedback a student received from cohort members or instructors.
Figure 4.22 States of overt behavior students went through

As a student started to solve a classwork problem s/he would start off in any of the three states: struggling, active and constructive. In Figure 4.22 this is indicated by the slanted arrow labeled as 1. As the student attempted to solve the problem s/he sought assistance from peer members of the cohort or from the TA/instructor of the class. Feedback from peers (indicated as slashed number 2 in Figure 4.22) also came in three forms (struggling, active or constructive). Feedback from TA/instructor was mostly constructive in nature. Based on the state a student was at and the type of feedback s/he received, the student could be either stuck at a struggling state, active or constructive state. For example, if a student was struggling and the cohort member whom s/he sought help from was also struggling, then both students were stuck in the struggling state. In this situation, the only way the students got out of struggling state was by asking help from the TA or the instructor.

In the context of solving a problem in this particular CT class, it was likely that students would be struggling or active in nature to start with before engaging in constructive or interactive behavior. This was because as new learners of CT, students needed time to learn the terminology and develop operational skills of the Computer Science discipline. Once this initiation was over and students had grasped terminologies and could implement computational actions, they could move to the next stage. The type of feedback a student received was based on the willingness to
help and prior knowledge of the group members. A student would be stuck in the struggling state if s/he did not receive active or constructive support when s/he was struggling. Similarly, when a student asked a shallow questions and receive a detailed explanation in response, the student can be thought to be exposed to deeper learning than initially initiated by the student.

Another result is that different cohorts exhibited different types of overt behaviors. The collective behavior of the cohort surfaced as a complex intertwined organism. In this study we saw three totally different types of dynamics happening in each of the three cohorts. Each cohort had a mix of students coming from different disciplines: College of liberal arts, College of Architecture and College of Science. The gender balance in each cohort was more or less the same. Still, we saw major differences in each cohort’s dynamics. In the Alpha cohort, where students were more or less at the same level of understanding, each trying to help out the other, successful at times, at times not. On the other hand, the Beta cohort had a member with some prior computing experience wanting to help his peers. We saw this student reaching out and helping other students of the cohort. And in Gamma cohort, where we have two students who didn’t interact with others, and, others who were at the same learning level, trying to help out each other as best as they could.

Socializing with cohort members seemed to help students bond with each other. In both Alpha and Beta cohorts we saw the presence of socialization episodes which were not present among the Gamma cohort. Also, being self-reflective was one overt behavior that was seldom seen except in a few students of cohort Beta. Prior experience in CS/programming, willingness to help others, communication and socialization skills, were important personal traits that all blended and meshed into cohort behavior.
CHAPTER 5 : UNDERSTANDING SOCIAL INTERACTIONS IN TERMS OF BEING ACTIVE VS. CONSTRUCTIVE: IMPLICATIONS ON LEARNING CT

5.1 Abstract

**Background:** There is limited research on different types of social interactions and their influence in learning Computational Thinking (CT).

**Purpose (Hypothesis):** The goal of this study is to test the hypothesis that the difference in type (active vs. constructive) of self-initiated social interactions exhibited by an online learner can lead to difference in learning of CT.

**Design/Method:** Scratch (www.Scratch.mit.edu) is an online community and social networking forum. In Scratch, one uses ‘blocks’, which are puzzle-piece shapes used to create programming code for solving a problem. Scratch has been designed to support the development of CT in young (ages 8 to 16). In the Scratch platform, one can code games, animations, and stories using a block-based programming language. Social interactions in this community are exhibited in the form of following another user or by commenting, remixing, and liking other users’ projects. By focusing on previously identified clusters of users (Gelman et al., 2016), this study characterized different types of social interactions as being either active or constructive on the basis of Chi’s (2009) Differentiated Overt Learning Activities (DOLA) framework.

**Results:** The findings of the study confirm the proposed hypothesis that in the context of Scratch users, difference in the type (active vs. constructive) of social interactions exhibited by a user lead to difference in learning of CT.

**Conclusions:** The analysis of social interactions and learning of CT revealed that there exists an association between active, constructive social interactions and learning CT. Users in different clusters exhibited a stronger relationship between active vs constructive social interactions.
5.2 Introduction

Scratch (www.Scratch.mit.edu) is an online community and social networking forum. In Scratch, one uses ‘blocks’, which are puzzle-piece shapes used to create programming code for solving a problem. The blocks connect to each other like a jigsaw puzzle, where each block represents a particular programming concept (e.g., if, do-if, repeat, end). Scratch has been designed to support the development of CT in young (ages 8 to 16) people (Brennan & Resnick, 2012; Dasgupta et al., 2016; Moreno-León et al., 2015; Robles, Moreno-León, Aivaloglou, & Hermans, 2017). Although the Scratch interface has been primarily designed for K-12 students, it also appeals to adults who have no previous knowledge of programming (Resnick et al., 2009). Scratch has been used in AP courses in high school and introductory programming courses at the college level (Bryant, Chinn, Hauser, Folsom, & Wallace, 2009; Ericson, Guzdial, & Biggers, 2007). Within Scratch, users have the opportunity to see projects completed by others, use pre-existing code, comment on each other’s projects and seek assistance from others. The social interactions in the Scratch online community take place around commenting, remixing, liking and sharing projects (Velasquez et al., 2014). These social factors of the Scratch community have been investigated in order to better understand the relational aspect of users’ social behavior and community involvement. Velasquez et al. (2014) utilized automated language coding to categorize comments into project related and other comments. The researchers found project comments to be richer in length, and included more verbs, adjectives and images than other comments. Scaffidi, Dahotre, and Zhang (2012) also studied comments and identified that certain types of comments (comments that included links to existing animations, provided details about the animation and an explicit request for particular feedback) lead to further discussion and collaboration in small groups. Fields et al. (2013) studied different trends of collaborative behaviors of Scratch community users. The authors in their study found that users who shared projects were more likely to engage in social activities like commenting on and favoriting others’ projects. Sylvan (2010) investigated social interactions in terms of project influence and social influence. Project influence was characterized as the number of times a project has been downloaded and social influence was calculated from betweenness centrality of friendship.
networks. Other social factors that Sylvan looked into were number of comments, number of times a user’s project had been featured and gallery participation etc.

Researchers have also looked into users’ experiences within Scratch and used the online community data to understand and assess the use of computational concepts demonstrated by the user (Brennan & Resnick, 2012; Dasgupta et al., 2016; Hermans & Aivaloglou, 2016; Hill & Monroy-Hernández, 2013; Manovich, 2005; Matias et al., 2016; Ota, Morimoto, & Kato, 2016; Yang et al., 2015). Brennan and Resnick (2012) and Matias et al. (2016) studied Scratch users’ project portfolios. By focusing on different types of the blocks used in projects, the studies (Brennan & Resnick, 2012; Matias et al., 2016) illustrate learning progression of CT concepts. On the other hand, Yang et al. (2015) operationalized informal learning in Scratch by calculating the inverse document frequency of each computational block and then cumulatively summing the weighted blocks of each learner. Hermans and Aivaloglou (2016) in their experimental study investigated users’ computational understanding in the presence and absence of duplicate and long codes. Robles et al. (2017) correlates the assessment of the CT skills of learners with copy and paste code present in learners’ project code. Moreno-León et al. (2015) and Ota et al. (2016) both separately developed automatic CT assessment tools for projects created by Scratch users. Some studies (Dasgupta et al., 2016; Monroy-Hernández & Resnick, 2008; Scaffidi & Chambers, 2012; Sylvan, 2010) investigated both the social interactions and the CT learning aspects of users of the Scratch community. Dasgupta et. al in their study focused on the association between remixing and learning CT (Dasgupta et al., 2016). The study used a number of remixes, downloads made by the user, experience of the user, comments received on projects created by the user, and number of different blocks used by the user as predictors. Similarly, based on social factors and differences in projects created, Scaffidi and Chambers (2012) categorized users as project leaders, active users, peripheral users and remixers/passive users. Monroy-Hernández and Resnick (2008) categorized users as active consumers, passive producers, and active producers.

The studies discussed above have investigated social interactions and learning of Scratch users from multiple perspectives. In most of the studies, CT learning activities of a user have been analyzed both in terms of actions initiated by himself/herself and/or actions performed by others that influenced the user’s learning and behavior.
In this study, we aim to focus only on actions initiated by the user. It is important to investigate such self-initiated social behaviors because these interactions are self-motivated and self-regulated by the learner himself/herself. Thus, in order to assess a learner’s overall experience in learning in an open online line platform it is necessary to focus on what s/he does as well as what s/he learns from the community. Figure 5.1 illustrates different social interactions initiated by a user in the context of the Scratch community. For example, User A in Figure 5.1 can leave a comment on a project, she can favorite projects created by other users, and follow other users in the Scratch community. User A can also create a project either by him/herself (original project) or by remixing code from pre-existing projects (remixed project). As a result the user is exposed to and learns CT concepts such as loops, conditions and operators. All these interactions: comment, favorite, follow and creating/remixing a project are initiated by User A.

Figure 5.1 Social interaction initiated by a user in the Scratch platform

The type of social interaction a user exhibits influences his/her learning. Some interactions cause deeper learning than others (Michelene Chi & Wylie, 2014). For example, simply copying and pasting code (active behavior) is less intense than modifying code and adding new features to a piece of code (constructive behavior). Chi’s active-constructive-interactive framework categorizes such social interactions according to the underlying cognitive value the interactions contribute towards learning.
Previous studies on Scratch have either solely looked at CT learning (Moreno-León et al., 2015; Yang et al., 2015), or at social interactions (Fields et al., 2013; Sylvan, 2010; Velasquez et al., 2014). Studies (Dasgupta et al., 2016; Matias et al., 2016) that have looked at the relationship between CT learning and social interactions have incorporated both self-initiated actions and the external social influence of community members. This study thus focuses on only self-initiated and self-motivated social interactions of Scratch users with the hypotheses that, the difference in the type (active vs. constructive) of self-initiated social interactions exhibited by a user can lead to difference in learning of CT.

5.3 Background and Theoretical Framework

This study is informed by three previous studies on Scratch community data. The three studies are: Gelman et al. (2016), Yang et al. (2015) and (Michelene Chi, 2009). This study will further explore users in clusters that have been already identified by Gelman et al. (2016). To evaluate CT learning, the study uses Yang et al. (2015)’s model of informal learning. To evaluate the social interactions of a user, Chi’s DOLA framework has been adapted. Figure 2 below illustrates the three studies that inform this study.

Figure 5.2 Studies informing this study
The following three sections provide an overview of the studies that inform the study and the theoretical model used in this study.

5.3.1 Clusters Identified by Gelman et al.

Gelman et al. (2016) in their study identified clusters of 8184 users of Scratch who had more than 25 followers. Gelman et al. used OpenOrd layout in Gephi to identify clusters of users. OpenOrd is a multi-level, force-directed layout and uses average-link clustering based on both edge weights and distance, where distance is determined using a force-directed algorithm (Martin, Brown, Klavans, & Boyack, 2011). Clusters of nodes were replaced by single nodes, and the clustering was repeated until a certain distance threshold between the nodes was reached. After the clustering was complete, the graph generated from the clustering algorithm was expanded by replacing the individual nodes with the original graphs in each cluster.

In these clusters, the authors explored the scratch age (the time a Scratch user remains active in the community) members and each group’s behavior in terms of the kind of projects the group created (e.g., games, cat videos, class project etc.). The authors found that specific interest-based subcultures were being formed within the Scratch community. The five clusters identified by the study were: a cluster with heavily featured old Scratch users (number of users 278), a young cluster of game makers (number of users 1798), a comparatively mature game making user cluster (number of users 2710), a cluster of users focusing on art projects and another which had a variety of projects (number of users 2260). Gelman et al. in their study did not explore the relationship between learning of CT and relevant social behavior of users within these clusters. The authors studied the interest based subcultures that emerged from the clustering algorithm using the lens of “Subcultural Theory of Urbanism” proposed by Fischer (1975).

This study further evaluates two of the clusters pre-identified by Gelman et al.; the cluster with comparatively mature game makers and the other cluster with variety of projects. For the purpose of this chapter we will identify these two clusters as Gaming cluster and Variety cluster. The reason for selecting two completely diverse clusters was to illustrate difference in cluster behavior. The Gaming cluster would exemplify comparatively keen CT learners and the Variety cluster would provide more of a general user behavior in Scratch.
5.3.2 Chi’s DOLA Framework

The nature of interactions during collaboration can lead to difference in learning. Certain types of actions lead to higher cognitive abilities. Chi’s DOLA framework suggests that interactions exhibited by learners can be categorized into three categories: active, constructive and interactive. An interactive interaction involves higher cognitive process than constructive, and constructive is higher cognitively than active. An active interaction can be noticed when a person does something physically or verbally. For example, when a student repeats a sentence verbatim or copies a problem’s solution steps (VanLehn et al., 2007). A constructive interaction is demonstrated when the output of the interaction goes beyond the information initially provided. For example, when a student explains his/her own understanding of a concept (M. T. Chi, Leeuw, Chiu, & LaVancher, 1994). An interactive interaction is exhibited between partners when both parties involved in collaboration equally contribute. For example, when two children jointly build a Lego model (Azmitia, 1988). Table 5.1 below provides definitions of active, constructive and interactive interactions and the underlying cognitive processes associated with them.

Table 5.1 Chi’s DOLA framework and cognitive process associated with each type of behavior

<table>
<thead>
<tr>
<th>Characteristics</th>
<th>Active</th>
<th>Constructive</th>
<th>Interactive</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Characteristics</strong></td>
<td>Doing something physically (e.g., following another user in an online community)</td>
<td>Producing outputs that contain ideas that go beyond the presented information (e.g., writing a piece of code without help of others)</td>
<td>Dialoguing substantively on the same topic, and not ignoring a partner’s contributions (e.g., equally contributing in writing a piece of code with another peer)</td>
</tr>
<tr>
<td><strong>Cognitive processes</strong></td>
<td>Attending Processes Activate existing knowledge Assimilate, encode, or</td>
<td>Creating Processes Infer new knowledge Integrate new information</td>
<td>Jointly Creating Processes Creating processes that incorporate a partner’s contributions</td>
</tr>
</tbody>
</table>
Classroom studies (Michelene Chi & Wylie, 2014; Menekse et al., 2013) using Chi’s DOLA framework have been used to evaluate students’ collaborative (between dyads or triads) behavior while learning concepts in physics, mathematics, bridge design, evolutionary biology, human circulatory system, introductory materials science and engineering. These studies have characterized selecting, repeating, paraphrasing, and manipulating an existing scenario in a simulation software as active behavior. On the other hand, when a learner elaborately explained a problem, made a connection to previous problems, generated a hypothesis, compared and contrasted, or drew analogies, the learner was considered to be constructive. An interactive interaction in collaboration was revealed when both partner’s debated each other’s ideas, when an instructor provided feedback which led to extended dialogue discussing the issue etc. This study operationalizes Scratch users’ social interactions in terms of being active and constructive and investigates their relationship to CT learning. This study did not consider interactive activities since the data used for the analysis of this study did not include collaborative/joint project related data.

5.3.2.1 Operationalization of Active and Constructive Social Interactions in Scratch™

This study defines social interactions of a user by counting the number of times the user initiated different types of interactions. For example; the total number of users a particular user follows, the number of times a user favorites other projects, the number of times a user goes out and makes comments on others’ projects, and the number of times a user remixes another user’s project. Figure 5.1 illustrates these interactions as comment, favorite, follow and remixing projects. In the case of adapting Chi’s DOLA framework, the study further classifies these interactions into active and constructive. Table 1 defines all three types of interactions. Interactions that do not modify or elaborate on the topic were considered active (e.g. follow,
In terms of categorizing comments as active and constructive, Velasquez et al. (2014)’s findings have been applied. According to Velasquez et al. (2014), comments with word count less than 18 (usually emojis, encouraging phrases, verbs and adjectives) were categorized as active and comments with word count more than 18 as constructive. However, in order to better understand the constructive nature of the comments, comments with greater than 18 words were manually coded by the researcher. Table 5.2 outlines the operationalized definitions of active interactions.

Table 5.2 Operationalized definitions of active interactions in Scratch

<table>
<thead>
<tr>
<th>Active Interactions</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Favorited</td>
<td>Total times a user favorites other projects</td>
</tr>
<tr>
<td>Follow</td>
<td>Total users a particular user follows</td>
</tr>
<tr>
<td>Active_Comment</td>
<td>Total times a user makes active comments on other projects. Active comments are defined to be comments that are less than 18 words (Velasquez et al., 2014)</td>
</tr>
<tr>
<td>Active_Remix</td>
<td>Total projects created by a user that was a remix of another user’s project where the number of different types of blocks uses is the same as the original project</td>
</tr>
</tbody>
</table>

Interactions that elaborated or incorporated users’ feedback were considered constructive (e.g. original projects, remixed projects with extra features). Table 5.3 provides definitions of constructive interactions in the Scratch learning community.

Table 5.3 Operationalized definitions of constructive interactions in Scratch

<table>
<thead>
<tr>
<th>Constructive Interactions</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original_Projects</td>
<td>Total number of original projects created by a user</td>
</tr>
<tr>
<td>Constructive_Comment</td>
<td>Total times a user makes a constructive comment on other projects Word count&gt; 18.320 and containing constructive praise or criticism. (Velasquez et al., 2014)</td>
</tr>
<tr>
<td>Constructive_Remix</td>
<td>Total number of projects created by a user that was a remix of another user's project where the number of different types of blocks used is more than the original project</td>
</tr>
</tbody>
</table>

5.3.3 Evaluating CT Learning using Yang et al.’s Model

In order to assess learning of CT of Scratch users, the Yang et al. (2015) model of informal learning has been used in this study. This model of learning does not just consider the total number of programming blocks used by a Scratch user but also accommodates how commonly or rarely a certain block is used in the community. Yang et al. in their model used Inverse Document Frequency (IDF) to assess informal learning in Scratch members. IDF is a widely used statistical measure used to illustrate how important a word is in a document (Manning, Raghavan, & Schütze, 2008; Sparck Jones, 1972). IDF assigns more weight to key words that appear rarely than to the ones that are more commonly used. In the context of Scratch data, instead of words, computational blocks were considered as the unit of analysis for IDF calculation. Yang’s model assigned higher weight to computational blocks that were rarely used and lower weight to blocks used frequently. Based on the different types of blocks used in original projects created by a user, Yang’s model calculates a cumulative value of learning. This model of evaluating learning is not ideal, but is a necessary proxy. The large scale of users make the implementation of more detailed forms of CT learning assessment impractical. Also, the researcher did not have direct access to the cluster users for any further investigation. Prior studies (Brennan & Resnick, 2012; Dasgupta et al., 2016; Moreno-León & Robles, 2015) have used block occurrence as a reasonable approximation, this could be assumed for our purpose as reasonably reflecting the efforts of users.

5.3.3.1 Adaptation of Yang’s Model in the context of this study

In Yang et al.’s study all Scratch users had at least 50 projects. The cumulative value of informal learning was calculated based on these 50 projects. In this study, all cluster members did not necessarily have 50 projects. This study selected users in the clusters with at least 5 projects. And the cumulative value was calculated based on all original projects created by a user. See appendix B for a detailed explanation of Yang’s model and appendix C for a sample calculation.
of how learning was calculated for a Scratch user. Also Yang et al.’s study used the term informal learning. This study refers to informal learning of a user to overall CT learning. We also subcategorize learning in three sets, loop learning, conditional learning and operator learning in the context of Scratch programming. Table 5.4 indicates the different blocks used to operationalize each sub learning category. Studies evaluating CT (Brennan & Resnick, 2012; Dasgupta et al., 2016; Moreno-León et al., 2015) in Scratch users have also used similar blocks to assess the different categories of CT learning.

Table 5.4 Sub categories of CT learning and corresponding blocks used to evaluate

<table>
<thead>
<tr>
<th>Type of learning</th>
<th>Blocks used</th>
</tr>
</thead>
<tbody>
<tr>
<td>Loop</td>
<td>forever, foreverIf, repeat, repeatUntil</td>
</tr>
<tr>
<td>Conditional</td>
<td>waitUntil, foreverIf, if, ifElse, repeatUntil, bounceOffEdge, turn-AwayFromEdge, touching, touchingColor, colorSees, mousePressed, key-Pressed, isLoud, sensor, sensorPressed, lessThan, equalTo, greaterThan, and, or, not, listContains</td>
</tr>
<tr>
<td>Operator</td>
<td>lessThan, equalTo, greaterThan, and, or, not, add, subtract, multiply, divide, pickRandomFromTo, concatenateWith, letterOf, stringLength, mod, round, abs, sqrt</td>
</tr>
</tbody>
</table>

By focusing on previously identified clusters of Scratch users (Gelman et al., 2016), this study investigates CT learning by slightly adapting Yang et al. (2015)’s model of informal learning. This study uses Chi’s DOLA framework to categorize social interactions initiated by a Scratch user as either active or constructive and investigates the relationship between differences in social interactions with learning of CT. The following section describes the research design of this study.

5.4 Research Design

The goal of this study is to test the hypothesis that difference in the type (active vs. constructive) of self-initiated social interactions exhibited by a Scratch user can lead to difference in learning of CT. In order to do so, the following data and analysis methods were used.
5.4.1 Data and Selection

The Scratch community datasets from 2007 through 2012 are publicly available to researchers for analysis. This contains data from the MySQL database that runs the Scratch online community website. There are 18 core datasets available for analysis that describe the major objects and relationships captured by the Scratch Online Community website. Appendix D provides a brief overview of the datasets and their properties.

The derived datasets of the clusters identified by Gelman et al. (2016) were also available to the researcher. This contains a dataset on users with at least 25 followers and the list of users (user id) belonging to each cluster. The Gaming cluster initially had 2710 users. However, only 2173 users had at least 5 projects. Similarly, the Variety cluster initially had 2260 users, of which 1933 users were used for analysis.

5.4.2 Data Processing

The collaborative learning platform of Scratch has been developed by educational researchers and computer scientists from MIT. Scratch community data has been used by a number of researchers (Brennan & Resnick, 2012; Fields et al., 2013; Hill & Monroy-Hernández, 2013; Velasquez et al., 2014) who have already published articles in renowned journals and conference proceedings. Thus, the data coming from Scratch is considered to be of good quality. Currently, Scratch has over 6 million projects shared and 4 million registered users and 33 million comments posted. The researcher had access to community data starting from 2007 to 2012. Relational data pertaining to the community were well defined and contain sufficient attributes for analysis. In order to extract social interaction and project related metadata of users of both clusters, the researcher used python scripts. Each user and project in the Scratch community was assigned a unique id (user_id and project_id). In order to create the project related profile of a user (e.g. number of original projects and number of remixed projects) the researcher extracted a total of 418,985 projects created by users by both the Gaming and Variety cluster from an observation of 1,928,699 project details provided by the Scratch community data. For each project, the researcher investigated the different types of blocks used, and compared the blocks used in the project with parent project blocks (if the project was a remix). In order to calculate
social interactions, e.g. follow, the researcher extracted the number of users a particular user followed. In order to calculate this value the researcher wrote python code to go through 1,313,200 observations of follower-followed dataset. Similarly, in order to calculate favorite projects the researcher had to go through 1,041,387 observations of data containing the relationship between project id and the user ids of users who had favorited that particular project. The final number of users, projects the researcher used for further analysis in this study is listed in Table 5.5 below.

Table 5.5 Cluster wise information regarding number of projects, comments etc., that was further analyzed in this study

<table>
<thead>
<tr>
<th>Name of Cluster</th>
<th>Total number of users (with at least 5 projects)</th>
<th>Total number of projects created by the cluster</th>
<th>Total number of comments made by users of the cluster</th>
<th>Total number of users the cluster users follow</th>
<th>Total number of projects favorited by cluster users</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gaming</td>
<td>2173</td>
<td>223,518</td>
<td>901,677</td>
<td>197,559</td>
<td>229,052</td>
</tr>
<tr>
<td>Variety</td>
<td>1933</td>
<td>188,759</td>
<td>677,641</td>
<td>185,447</td>
<td>144,475</td>
</tr>
</tbody>
</table>

5.4.3 Analysis

In order to test the hypothesis that difference in the type (active vs. constructive) of self-initiated social interactions exhibited by a Scratch user can lead to difference in learning of CT, the following three steps of analysis were carried out in this study. First: CT learning was calculated for each user. Second, for users of both clusters, different interactions were identified and counted. Third, correlational analysis was conducted between different active and constructive interactions with learning. Correlational analysis is suitable for identifying the relationships between two or more variables, in order to understand how the variables change together (Howell, 2012). These three steps allowed us to examine the one-to-one relationship between different types of social interactions and CT learning.
5.5 Findings

This section is subdivided into three sections. The first section presents the calculated learning scores of users in both clusters. The second section describes the calculated values of social interactions. The final section illustrates the relation between the social interactions and learning of CT.

5.5.1 Learning Scores of Users in chosen Clusters

Figure 5.3 illustrates the comparison of calculated overall CT scores of users of the Gaming (number of users 2173) and the Variety cluster (number of users 1933). The maximum CT learning score (318.54) of a user of the Gaming cluster was higher than the maximum score (312.37) of the Variety cluster. Users of the Gaming cluster scored higher in CT learning than the Variety cluster users. Nearly 50% of the Gaming cluster users’ CT learning scores were in the range between 100-200, compared to only 29% the Variety cluster users’ scores. 70% of the Variety cluster users’ CT learning scores were less than 100, compared to 49% of users of the Gaming cluster. The average CT learning score of users of the Gaming cluster was 99.1 and the Variety cluster was 73.60 (See Table 5.6).

Figure 5.3 Comparison of overall learning score between Gaming and Variety cluster users
Further categorical analysis of the overall CT learning into Loop, Conditional and Operator learning was calculated for users of the Gaming and Variety cluster. This subcategorization of overall learning allows us to focus on important CT concepts, loops, conditions and operators, separately and assess if users in both clusters are learning all three concepts equally. Figure 5.4-Figure 5.6 illustrate the comparison of these scores between the two clusters. The maximal value (when a user had used all the blocks identified to fall under the Loop category, see Table 5.4) a Scratch user can score on Loop learning was 4.19. In terms of Loop learning score (see Figure 5.4), 70% of the users of the Gaming cluster scored higher than 4, compared to 48% of the users of the Variety cluster. For both the Gaming and Variety cluster, roughly 17% scored in the range of 2.99-4 and only 12% of the Gaming cluster users scored less than 2.99, compared to 35% of the Variety cluster. The average Loop learning score of a user of the Gaming cluster was 3.66 and the Variety cluster was 2.97 (see Table 5.6). Thus, overall users of the Gaming cluster had a better score on Loop learning than users of the Variety cluster.

Figure 5.4 Comparison of loop learning score between Gaming and Variety cluster users

Figure 5.5 illustrates the comparison between Operator learning scores of the Gaming and Variety cluster. The highest score a user from both clusters obtained in Operator learning in this study was 41.39. From Figure 5.5 Comparison of operator learning score between Gaming and Variety cluster users, we see that 55% of the users of the Gaming cluster scored more than 20 in Operator learning, whereas 57% of the users of the Variety cluster scored in the range of 10-20. The average Operator learning score of a user in the Gaming cluster was 12.43, and 8.89
for a user of the Variety cluster. Thus, in terms of Operator learning, users of the Gaming cluster had a better score than users of the Variety cluster.

Figure 5.5 Comparison of operator learning score between Gaming and Variety cluster users

Figure 5.6 illustrates the comparison between Conditional learning scores of the Gaming and Variety cluster. The highest score a user from both these clusters obtained in Conditional learning in this study was 31.48. From Figure 5.5 Comparison of operator learning score between Gaming and Variety cluster users, we see that 46% of users of the Gaming cluster scored more than 20 in Conditional learning, compared to 27% of users of the Variety cluster. However, for Conditional learning scores between 10-20, the percentage count of users of the Gaming (44%) and Variety (42%) cluster were about the same. The average Conditional learning score of a user in the Gaming cluster was 18.5, and for a user of the Variety cluster 14.3. Thus, in terms of Conditional learning also, users of the Gaming cluster had a better score than users of the Variety cluster.
Thus, from the CT learning perspective of this study, users of the Gaming cluster scored higher than users of the Variety cluster, overall as well as in the sub categories of Loop, Operator and Conditional learning. A possible explanation of why users of the Gaming cluster learned more compared to users of the Variety cluster is the difference in the nature of projects created by the two clusters. Game design in the Scratch platform has been found to be effective in fostering CT in young children (Repenning et al., 2010). Design and implementation of the simplest of the games requires the user to programmatically implement some form of winning condition, and create multiple opportunities for a player to take. This requires the designer of the gaming project to explore and use a diverse range of computational blocks from the Scratch platform, compared to art or story telling projects (typical projects created by users of the Variety clusters) which typically involve a limited set of programmable blocks.

5.5.2 Calculated values of Social Interactions

In terms of social interactions, Figure 5.7 illustrates the average number of different active (e.g. follow, comment, favorite) interactions of the Gaming cluster and Variety cluster. In terms of follow (Gaming: 91, Variety: 95) and active remix the average scores (17 for both clusters) are close to each other. The average number of projects favorited by users of the Gaming cluster
(105) is much higher than the average number (80) of the Variety cluster. In terms of average number of active comments, the Gaming cluster was higher (415) than the Variety cluster (352).

Figure 5.7 Comparison of average active social interactions between Gaming and Variety cluster

Figure 5.8 illustrates the average number of constructive interactions of both clusters. In terms of constructive interactions, we did not notice a significant difference between the Gaming and the Variety cluster. Average original project was 71 for both clusters and constructive remix was 14 for the Gaming cluster and 10 for the Variety cluster. Overall, in terms of average number of social interactions exhibited by users of both clusters, there were no extreme differences, excluding the average number of projects favorited and comments made on projects, which were both higher in value for the Gaming cluster.
The summary statistics for measures of social interactions and learning of users of the Gaming and the Variety cluster are provided in Table 5.6. In terms of differences between clusters, values for favorites was higher in the Gaming cluster than the Variety cluster. All other social interactions values were more or less in the same range for both the Gaming and Variety cluster. In terms of learning, the Gaming cluster had higher values (mean, median and maximum) than the Variety cluster.
Before correlating social interactions with learning, some factors (e.g., remix_active and constructive, follow, favorited) were log-transformed to achieve normality. This was done because those particular data sets were highly skewed. Previous studies (Hill & Monroy-Hernández, 2013; Manovich, 2005) on Scratch data (e.g. number of remixed projects of a user and comments made on a user’s projects were log transforms before being used in a predictive regression model) have also used log-transformed data sets for analysis.
5.5.3 Relational between Learning and Social Interactions

Table 5.7 shows the Pearson correlation coefficient values corresponding to analysis of different measures of social interactions with learning of CT. A Pearson correlational coefficient value less and equal to .2 was considered low, between .5-.3 was considered moderate, and more than .6 was considered highly correlated. In terms of the relationship between Total active interactions and Learning of CT, Pearson’s correlational analysis revealed a moderate positive correlation r=.326, p<.01 for Total CT learning and lower positive correlation r=.261 for Loop learning, r=.293 Conditional learning, r=.264 Operator learning for users of the Gaming cluster. This means that a user of the Gaming cluster who exhibited more active interactions also learned more. Similar but slightly lower Pearson correlational values between Total active social interactions and learning of CT was found for the users of the Variety cluster (moderate positive correlation r=.315 for Total CT learning). The results of the correlational table (table 6) also show Pearson coefficient values for the analysis of different categories of active social interaction (e.g. follow, favorite, remix_active and comment) with learning of CT. Amongst these values, remix_active had a higher- moderate positive relationship (r=.472) with Total CT learning for the Gaming cluster compared to the Variety cluster (r=.466), which was also moderately positive. In terms of the relationship between comments and learning, the Variety cluster had a higher-moderate positive relationship (r=.351 for Total CT learn and r=.278 for Loop, r=.297 Conditional and r.282 Operator Learning).
Table 5.7 Summary Pearson correlation analysis of social interaction factors and CT learning of the Gaming cluster (columns include values of Pearson correlation coefficient)

<table>
<thead>
<tr>
<th></th>
<th>Total CT Learning</th>
<th>Loop Learning</th>
<th>Conditional Learning</th>
<th>Operator Learning</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Active Interactions</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>log_follow</td>
<td>Gaming .078**</td>
<td>.067**</td>
<td>.072**</td>
<td>.070**</td>
</tr>
<tr>
<td></td>
<td>Variety .024</td>
<td>-.026</td>
<td>.002</td>
<td>.019</td>
</tr>
<tr>
<td>log_comment_active</td>
<td>Gaming .293**</td>
<td>.252**</td>
<td>.259**</td>
<td>.221**</td>
</tr>
<tr>
<td></td>
<td>Variety .351**</td>
<td>.278**</td>
<td>.297**</td>
<td>.282**</td>
</tr>
<tr>
<td>log_favorited</td>
<td>Gaming .228**</td>
<td>.199**</td>
<td>.228**</td>
<td>.191**</td>
</tr>
<tr>
<td></td>
<td>Variety .209**</td>
<td>.176**</td>
<td>.182**</td>
<td>.178**</td>
</tr>
<tr>
<td>log_remix_active</td>
<td>Gaming .472**</td>
<td>.336**</td>
<td>.448**</td>
<td>.426**</td>
</tr>
<tr>
<td></td>
<td>Variety .466**</td>
<td>.390**</td>
<td>.438**</td>
<td>.419**</td>
</tr>
<tr>
<td>Total_log_active</td>
<td>Gaming .326**</td>
<td>.261**</td>
<td>.293**</td>
<td>.264**</td>
</tr>
<tr>
<td></td>
<td>Variety .315**</td>
<td>.237**</td>
<td>.266**</td>
<td>.259**</td>
</tr>
<tr>
<td><strong>Constructive Interactions</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>log_original</td>
<td>Gaming .466**</td>
<td>.312**</td>
<td>.404**</td>
<td>.430**</td>
</tr>
<tr>
<td></td>
<td>Variety .545**</td>
<td>.450**</td>
<td>.507**</td>
<td>.483**</td>
</tr>
<tr>
<td>log_remix_constructive</td>
<td>Gaming .209**</td>
<td>.143**</td>
<td>.178**</td>
<td>.182**</td>
</tr>
<tr>
<td></td>
<td>Variety .636**</td>
<td>.533**</td>
<td>.602**</td>
<td>.588**</td>
</tr>
<tr>
<td>Total_log_constructive</td>
<td>Gaming .497**</td>
<td>.337**</td>
<td>.459**</td>
<td>.433**</td>
</tr>
<tr>
<td></td>
<td>Variety .578**</td>
<td>.481**</td>
<td>.540**</td>
<td>.516**</td>
</tr>
</tbody>
</table>

**Significant at the 0.01 level (2-tailed), Red value indicates higher value amongst the two cluster

In terms of Pearson coefficient values between Total constructive interactions and CT learning, the Variety cluster had a higher moderately positive correlation (r=.578 for Total CT learning, r=.481 for Loop, r=.540 for Conditional and r=.516 for Operator learning) compared to the Gaming cluster (r=.497 for Total CT learning, r=.337 for Loop, r=.459 for Conditional and r=.433 for Operator learning). In terms of the relationship between categorized constructive social interactions: original projects and remix_constructive and learning of CT, the Variety cluster had stronger correlational values compared to the Gaming cluster. In terms of most salient correlational coefficient values, the relationship between Variety cluster’s remix_constructive and Total CT learn was a strong positive correlational value of r=.636 compared to the Gaming cluster’s r=.209.
Overall, in terms of Total active vs. Total constructive social interactions, Total constructive social interactions had a stronger relationship with Total CT learning (Variety r= .578, Gaming r =.497) than Total active interactions with Total CT learning (Gaming r=.326, Variety r=.315) for both clusters. This means that users of both clusters learned more by creating original projects and extensively remixing (remix_constructive) projects than by just following, commenting and simply remixing (remix_active) code. This finding that constructive behavior leads to greater learning confirms previous studies that have investigated active vs. constructive behavior of learners. However, in the case of constructive interactions, the Variety cluster had stronger positive relationships with learning, and in case of active interactions, the Gaming cluster had stronger positive relationships with learning.

Since statistically significant correlations do exist between active and constructive interactions and learning for both of the selected clusters, it can be said that there is a relationship between active/constructive social interactions and learning of CT, which confirms the proposed hypothesis of this study.

5.6 Discussion

The findings of the study confirm the proposed hypothesis that in the context of Scratch users, difference in the type (active vs. constructive) of social interactions exhibited by a user can lead to difference in learning of CT. For example, a user who had created more original projects and more remix projects with added features to the existing code learned more than a user who did less of either of these two constructive activities. Creating remixed projects and commenting on projects were also found to be active behavior that correlates to learning of CT. In terms of separate CT concepts, including loop, conditions and operators; Conditional learning correlated higher with constructive interactions than the Loop or Operator learning. For both clusters, learning from constructive interactions was found to be stronger than the relationship between CT learning and active interactions. In terms of difference between clusters and relationship between social interactions and learning CT, users of the Variety cluster learned more by creating original projects and extensively remixing (remix_constructive) projects than users of the Gaming clusters. However, when it came to active interactions, users of the Gaming cluster learned more by just following, commenting and simply remixing (remix_active) code than users of the Variety cluster. A stronger association of active interactions with the Gaming clusters and
that of constructive interactions with the Variety clusters could be explained by the difference in the learning scores of the users in both clusters. Since users of the Gaming cluster scored higher compared to users of the Variety cluster, it could be that the users of the Gaming cluster can learn more just by casually investigating (favoriting/active_remix) projects, whereas the users of the Variety cluster need to explore the project in depth (constructive_remix) in order to understand and use a computational concept.

5.6.1 Implications for Practice

CT educators should encourage active and constructive behavior in novice learners. Novice learners can start off learning by being active: remixing codes written by others, and casually discussing project related topics with CT learners in the community. Gradually, the users should be encouraged to add new features to pre-written code (the idea of remixing with added features) and keep on creating new computational projects. Novice learners should also be encouraged to discuss project features and provide constructive feedback during group discussions.

Instead of just looking at how much a user has learned, the framework of this study provides an overall understanding of a user in terms of being active and constructive. According to Chi (2009), constructive behavior suggests deeper learning than active behavior. In the context of open online learning, it is difficult to assess how much a user has actually learned. Looking at learning in terms of being active and constructive provides an improved understanding of the user’s learning and the influence of community interactions on learning.

Tools that have been designed to assess CT learning (Moreno-León et al., 2015; Ota et al., 2016) of users are primarily focused only on the CT learning aspect. The framework used in this study enables us to better assess the collaborative nature of a user. The assessment of social interactions can be used to provide guided feedback to the user. As we see more and more open online environments for learning to code, the community aspects of learning are becoming more and more important. It is crucial for learners to be proactive in their own learning and seek out assistance. Self-assessment tools are also becoming crucial for learners to gauge on their own learning and opportunities for improvement.
5.6.2 Limitations and Future Work

The Scratch data set has one million user data available for analysis. The data set in this study comprised of 4000 users of this data. Thus, the findings are limited to the particular data set. The active and constructive notion of social interactions can be applied to the entire data set. Also, other social factors e.g., scratch age, gallery participation etc. can also be used to further analyze user behavior and learning of CT.

5.7 Conclusion

In this study, an approach was presented for evaluating social interactions initiated by users in the Scratch community. Chi’s DOLA framework has been adapted to characterize online behavior that suggests deeper learning. Interactions such as following, favoriting and remixing exact project code were characterized as active. Interactions such as original projects created and remixed projects that add features were considered to be constructive. Although comments were also characterized as active and constructive, due to the nature of the community users, constructive comments were scarcely observed. The analysis of social interactions and learning of CT revealed that there exists a relationship between social interactions (active, constructive) and learning of CT. Users in different clusters exhibited a stronger relationship between active vs constructive social interactions. For example, in this study it was found that the users who are more interested in developing games were seen to learn more just by casually investigating (favoriting/active_remix) projects. On the other hand, the users of the Variety cluster, mostly creating art and story projects, needed to explore the projects in depth (constructive_remix) in order to understand and use a computational concept.
CHAPTER 6: DISCUSSION

The goal of this doctoral study was to better understand the collaborative aspects (particularly social interactions) of learning Computational Thinking (CT). In order to do so, this dissertation study focused on answering the following three research questions:

RQ1: How do novice learners/non-CS majors describe their experiences of learning CT in a general education course?

RQ2: How do novice CT learners naturally interact with group members while learning CT in a classroom setting?

RQ3: Do social interactions exhibited by novice CT learners in an online learning platform lead to a difference in learning of CT?

These research questions aimed to investigate different but related aspects related to collaboratively learning CT by novice learners. RQ1 provides an overall understanding of the contextual factors and characterizes collaborative aspects of learning in a CT face-to-face classroom at a large Southeastern University. RQ2 investigates the social interaction occurring between group members of the same classroom. And RQ3 focuses on the relationship between different social interactions initiated by users and learning of CT in an online learning platform Scratch™. First, a summary of the findings of the each of the research questions is presented in section 6.1. Then, these findings are synthesized in section 6.2. Section 6.3 highlights the practice implications and section 6.4 focuses on the research implications of this dissertation study.
6.1 Summary of Findings

A summary of the findings pertaining to each research question is presented in Table 6.1

Table 6.1 Research questions, data, and summary of findings

<table>
<thead>
<tr>
<th>Research Question</th>
<th>Data</th>
<th>Findings</th>
</tr>
</thead>
</table>
| RQ1: How do novice learners/non-CS majors describe their experiences of learning CT in a general education course? | Individual interviews of students majoring in different disciplines in a CT class at VT | • Learners from different disciplines had varied experiences in learning CT  
  • Students valued the cohort model of learning CT collaboratively  
  • Disciplinary datasets allowed students to combine their disciplinary knowledge with computational skills learned from the CT class |
| RQ2: How do novice CT learners naturally interact with group members while learning CT in a classroom setting? | Video Observations of a small group of CT learners in a CT class | • While solving a particular CT related problem, the novice learner transitioned through stages such as struggling, being active and being constructive  
  • The type of feedback a student received was based on the willingness to help and prior knowledge of the group members  
  • Different cohorts exhibited different types of overt behaviors. The collective behavior of the cohort surfaced as a complex intertwined organism. |
| RQ3: Do social interactions exhibited by novice CT learners in an online education context lead to learning CT? | Quantitative analysis of data collected by the [research method] | • Different types of social interactions (active vs. constructive) lead to difference in learning CT.  
  • Constructive social interactions lead to more CT learning |
learning platform (e.g., Scratch™) lead to difference in learning of CT?

| Scratch learning platform | • Users in different clusters exhibited a stronger positive relationship between active versus constructive social interactions and learning of CT |

6.1.1 Summary of findings of RQ1

Investigating students’ experiences through Stahl’s collaborative framework helped identify various factors that influenced learning CT of a multi-disciplinary group of undergraduate students. For most of the non-science majors, learning to code was learning something conceptually different than they were used to. These students struggled to write code, finding it difficult to comprehend how precise one had to be. Nevertheless, most of the students could comprehend the value of learning CT and its applicability to their own discipline. Most of them found the course instructor, teaching assistants and the long term cohort model to be valuable resources in learning CT. Asking and explaining problems to cohort members was not only useful to advance through a problem but also valuable to one’s own learning process. Discussing problems with students coming from different disciplines allowed them to see how diversely students perceive and explain a problem. Students coming in with a science background could recognize the struggle non-science majors faced while learning to code. Providing ample time for students to actively learn CT during class time, acknowledging that students will make mistakes, allowing students to casually communicate with each other, and overall structure of the course were a few salient features of the CT course that students brought up during their interviews.

6.1.2 Summary of findings of RQ2

In a class focused on introductory CT for non-CS-majors, forming long term collaborative groups/cohorts was a beneficial teaching strategy. The cohorts allowed students to receive a first line of support, creating a community where students from different disciplines could share their understanding of the new concepts and tools and as well voice their difficulties. The study used an adapted version of Chi’s DOLA framework in the context of categorizing different overt behaviors exhibited by novice learners. Analysis of these overt behaviors revealed
that novice learners of CT in this class, while solving a particular problem, transitioned through stages such as struggling, being active or being constructive (see Figure 6.1). The transition process and its outcomes were influenced by the student’s prior knowledge in CT, and the type of support and feedback a student received from cohort members or instructors. Figure 6.1 illustrates the different states that a student can be at: struggling, active or constructive (labeled as number 1). Based on the current state of a student and the type of feedback (which can also be struggling, active or constructive labeled as number 2 in Figure 6.1) s/he received the student is either stuck in a struggling state, or transitions to an active or constructive state.

The adapted coding framework characterizing students’ naturally occurring overt CT learning behavior further subcategorized the active and constructive behaviors of students.

![Diagram](image)

**Figure 6.1 States of overt behavior students went through**

Of all observed behaviors 87% were accounted for by just five behaviors: asking a shallow question, showing/looking at another student’s solved problem, asking an elaborate question, explaining to others and struggling on a task. Figure 6.2 illustrates the percentage count of the most observed interactions.
The findings of the research study answering RQ2 also suggest that different cohorts exhibited different types of overt behaviors. The collective behavior of the cohort surfaced as a complex intertwined organism. The researcher saw three totally different types of dynamics happening in each of the three cohorts.

6.1.3 Summary of findings of RQ3

In this study, Chi’s DOLA framework was adapted to characterize social interactions initiated by users in the Scratch community. Interactions such as following, favoriting and remixing exact project code were characterized as active. Interactions such as original projects created and remixed projects that added features were considered to be constructive. Although comments were also characterized as active and constructive, due to the nature of the community users, constructive comments were scarcely observed. The analysis of social interactions and learning of CT revealed a relationship between active and constructive social interactions and learning of CT. Constructive interactions had a higher relationship with learning of CT for users in both clusters. However, users in different clusters exhibited a stronger relationship between active versus constructive social interactions. Users of the Variety cluster had a stronger relationship between CT learning and constructive interactions compared to Gaming cluster
users, whereas the users of the Gaming cluster had a stronger relationship between CT learning and active social interactions than users of the Variety cluster.

6.2 Synthesizing Findings

Synthesizing the findings of the three studies, the researcher found that the following five salient features emerged:

6.2.1 The combination of Stahl’s and Chi’s frameworks as a lens to investigate ‘collaboratively learning CT’ in a cohort model was found to be useful for understanding the learning of novice learners

6.2.2 Adaptation of Chi’s framework provided a better understanding of the social aspect of collaboratively learning CT

6.2.3 Students’ disciplinary background/interest influences the learning process and the extent of learning CT

6.2.4 Novice CT learners displayed different stages of learning outcomes (struggling/active/constructive) while solving a problem

6.2.5 A supportive collaborative learning environment is important for novice CT learners in both face-to-face and online settings

The following sections elaborate on these five salient features.

6.2.1 The combination of Stahl’s and Chi’s frameworks as a lens to investigate ‘collaboratively learning CT’ in a cohort model was found useful for novice learners

Stahl’s collaborative framework helped the study (answering RQ1) to focus on the factors that influenced learning of CT in small groups. Further, the study (answering research question RQ2) using Chi’s DOLA framework illustrated the interactions between members of these small groups. Together, the factors that influenced learning and characteristics of interaction, can facilitate the identification of the salient features of small groups of non-CS majors collaboratively learning CT in a classroom setting. Figure 6.3 illustrates the combined outcomes of Stahl’s collaborative and Chi’s DOLA frameworks.
It is worth mentioning here that Stahl had developed the “influences of collaborative learning framework” (Stahl, 2010) as a lens to reflect on the complex dynamics of different factors that influence group cognition in collaborative learning. Stahl considered the task and outcome to be collaborative in nature. The term collaborative in this case suggests a common task or goal to be achieved by group of people. For RQ1 of this dissertation, the problem each group member was attempting to solve was not collaborative as suggested by Stahl. Rather, each student was solving the same individual task. The goal of collaboration was not to come up with a single solution but for each student to be able to solve a problem individually. Thus, the aim of collaboration was to learn from each other. However, Stahl’s framework was still found to be applicable in this particular context of collaborative learning since it captured the constraints/factors that influenced learning in small groups in general. Analysis of students’ experiences in learning CT in this particular general education course allowed the researcher to distinguish the role of factors such as background, context of interactions, students’ roles in small groups and individual outcomes. However, the field ‘team knowledge artifact’ of Stahl’s framework remained incomplete.
Also, adapting Chi’s framework in the context of learning CT helped the study focus on the observable behaviors exhibited by members within small groups. Findings of this part (Chapter 4) of the dissertation study revealed that observable behaviors were mostly active in nature and at times, students tended to get stuck in a struggling mode. This happened for students who were taking a CT related course for the first time. Students’ observable behaviors, such as verbal and physical interactions with each other, were considered. In the context of learning CT, these interactions were characterized as struggling, active and constructive according to Chi’s adapted framework (see figure 6 in section 4.5).

However, once findings from both studies were combined, the question that came up for consideration was: what should be considered as the “team knowledge artifact” of Stahl’s framework?

Students in this study were not collaboratively solving a single problem. Here, a distinction between solving a CT problem collaboratively versus collaboratively learning CT is made. Solving a CT problem collaboratively was defined as solving a single problem together in a group. And collaboratively learning CT was defined as spontaneously helping group members learn CT concepts. In the context of the class considered in this study, the term collaboratively learning CT was used.

Thus, the dialogue patterns that emerged from collaboratively learning CT could be considered as the team knowledge artifact. This is because the small group as a whole contributed to each member’s learning, by providing explanations, sharing problem solving strategies with one another or simply struggling together. As a result, the artifact or outcome of each group was considered to be the cumulative effect of different types of dialogue/interactions students had with each other. Michelene Chi and Menekse (2015) suggest that certain types of dialogue/interaction patterns promote greater learning than others. According to the authors, when a student initiates an active interaction and in response receives a constructive response (we refer to this interaction pattern as active -> constructive), the student learns more compared to when the response is active in nature (the interaction pattern would be active -> active). Following Chi and Meneske’s interaction pattern framework (2015), each interaction between two students of the CT class fell under one of the five categories illustrated in Figure 6.4. It can be noted that most interactions initiated by students in the CT course were active in nature.
However, some responses were active (33%) and some responses were constructive (32%) in nature.

![Dialog Pattern observed between CT students]

Figure 6.4 Interaction pattern exhibited by all observed cohort members

Thus, the incorporation of Stahl’s “influences of collaborative learning framework” (Stahl, 2010) and Chi’s “Differentiated Overt Learning Activities” (Michelene Chi, 2009) framework was able to illustrate student’s experiences of collaboratively learning CT in a cohort model.

6.2.2 Adaptation of Chi’s framework provided better understanding of the social aspect of collaboratively learning CT

Chi’s adapted framework, in the context of the classroom setting, and the classification of social interactions, in the online setting (Scratch), was used to evaluate novice learners’ behavior in collaborative settings, indicating the quality of the social interactions. CT assessment methodologies in research studies and classroom settings have mostly focused on the measuring concepts relating to CT online, leaving out the social aspects. However, most of the operationalized definitions of CT in literature (see Chapter 2) have some form of collaborative aspect interwoven in them. Thus, in terms of assessment of CT it is also appropriate to assess collaborative aspects of CT learning as well. The assessment methodology (for both online and
face-to-face settings) coming out of this dissertation study captures the collaborative aspects of CT learning of an individual learner, which is also suggestive of the cognitive level associated with such collaborative behavior.

6.2.3 Students’ disciplinary background/interest influences learning process and extent of learning CT

The discipline a student came from had a profound influence on how the student perceived learning CT. From the interview and observational data, it was evident that students majoring in History, Political Science, Public Relations, and English initially found learning CT to be different than what they were used to learning in their own discipline. Typically, this group of students took courses where the emphasis was on writing answers, reading essays and not having an absolute right or wrong answer for assignments. However, in the CT course these students found the need to be analytical (solving a problem by first breaking it down into parts) and to be precise (the need to follow the syntax and semantic of a programming language). The computer science terminologies were new to them. They also got confused when they saw the same thing being solved in multiple ways. On the other hand, a student coming from a science background found learning CT to be typical. The student emphasized the fact that he usually had an analytic approach towards learning, and because of that learning CT was not different for him.

In the Scratch study, the author found users with differences in interests indicating a difference in the way they socially interacted in the community. Users who were mostly into creating Games (the Gaming cluster) indicated a stronger positive relationship between active social interactions (e.g. following, commenting) and learning of CT than users who created various types of projects e.g., storytelling projects, gaming, coloring projects etc. (Variety cluster). Thus, the background and the way one is used to tackling problems have a significant influence in one’s approach towards learning CT.
6.2.4 Novice CT learners displayed different stages of learning outcomes (struggling/active/constructive) while solving a problem

The findings of the observational study of the CT course (at VT) suggest that this particular group of students, mostly majoring in a discipline in liberal arts, were active in nature while communicating with each other. For most of the students, the CT class was the first computer science/programming related course they had ever taken. The terminologies were new and the way of thinking was new too. While interacting with each other, instead of articulating the problem, they would rather show their computer screen and look at someone else’s solved solution – all of which were active in nature. Similarly, in the Scratch community users were seen to follow other users in the community, like and remix projects created by others, and make comments; all of which was considered active behavior by the user. The findings of the Scratch study suggest that there is a positive relationship between active interactions and learning of CT.

The findings of the classroom observational study also exemplified the ‘struggling’ state of a novice CT learner. A struggling state was defined as a state where two or more students did not know how to solve a problem and were going back and forth on ideas, or randomly trying out different options of the software. External feedback was crucial in getting students out of this struggling state. However, in the Scratch learning platform, the researcher could not capture such struggling behavior of a user.

6.2.5 A supportive collaborative learning environment is important for novice CT learners in both face-to-face and online settings

Students in general found that being part of a group where one can ask questions while learning CT is very helpful. Assigning students into small interdisciplinary groups for the entire semester allowed students to get to know their classmates. They were able to see how students from different backgrounds solved problems, and described computational concepts. The temperament of the instructor and TAs was also important for students. They found that having an instructor who acknowledged and accepted that students could find learning CT difficult helped them stay motivated in the class. Along with qualitative interview data, a survey (see Figure 6.5) administered during the middle of the semester also confirmed that students found
the cohort, instructor, and TA helpful towards their learning of CT. The response rate of the survey was 83%.

**Results of survey administered during the middle of the semester**

![Survey Results Graph]

Questions were asked in the form of “Do you strongly agree, agree, disagree, or strongly disagree with the following statement?”

Survey respond rate 83%

Figure 6.5 Results of survey administered during the middle of the semester

In the Scratch study, the researcher also found users of the online community to be supportive of each other. Users extensively interacted with other users in the community. Scratch users are able to re-use and modify projects created by other users, and ask for help from others by posting comments. These interactions have a positive influence on how the user learns CT. Thus, in both the setups studied in this work, an online platform and a college classroom, it is important to foster a supportive learning community for CT learning.

6.2.6 Summary of Synthesis

Synthesizing findings coming from the interview and observation based study of the face-to-face classroom setup (chapter 3 and 4) and the quantitative study of the online CT learning platform (chapter 5) allowed us to better understand the collaborative and behavioral aspects of a novice CT learner, and gauge the importance of collaborative CT learning.
6.3 Practice Implications

The findings of this research facilitated in formulating the following recommendations for educators regarding the design and assessment of CT related problems:

6.3.1 Allow students to go through different kinds (active/constructive) of problems:

Educators should design problems that foster active/constructive behavior in novice CT learners. Novice learners can start off by solving active problems such as examination of pre-written codes by experts, or making minor changes to pre-existing codes. Gradually, the learners should be encouraged to add new features to pre-written code (the constructive idea of remixing with added features) and keep on creating new computational projects. Novice learners should also be encouraged to discuss project features and provide constructive feedback during group discussions.

6.3.2 Provide students with the opportunity to work with disciplinary/interest based datasets while learning CT:

Working on disciplinary/interest based data sets can help students reflect on the computational concepts they are learning and start applying these concepts in their own disciplines. For example, an English major, while computationally analyzing Amazon comments, found that just searching for negative key words in comments was not the only indication of a negative remark. Rather, she goes on suggesting features that could be incorporated to increase the accuracy of such sentiment analysis tools.

6.3.3 Keep in mind, the learning curve is steeper for certain learners:

Educators should keep in mind that learners with no prior programming experience and those who are not used to the process of breaking problems into computationally independent sub pieces may find it rather difficult to begin the problem solving process. Also, the syntax and semantics of writing code can be daunting for some students who are not used the precision involved in writing a piece of code.
6.3.4 Provide students a collaborative space to learn CT with others:

Novice learners can find learning CT to be difficult. Thus, it is important to create a relaxed, supportive learning environment that fosters CT learning and retains novice CT learners. A collaborative environment either in class or in an online setting can be beneficial for novice learners. A collaborative environment allows students to see what other learners are working on, ask for help when needed, and also gain an opportunity to gauge one’s own learning.

6.3.5 Distribute students with prior programming experience:

In forming teams for in-class team work it is beneficial to distribute learners with prior programming experience to different cohorts so that each cohort has at least one expert to serve as a means to help other students in the cohort.

6.3.6 Take a social-cultural approach in evaluating CT learning:

Taking a social-cultural approach in evaluating CT learning can help the educator gauge not only the level of understanding of the learner, but also gain a behavioral picture of the learner. The collaborative aspects of learning CT are frequently reported in CT literature. It is also important that educational settings give similar treatment to such learning behaviors. The adapted framework in this dissertation (chapter 4) can assist in evaluating such collaborative behavior of novice CT learners.

6.4 Research Contributions

The following list highlights the research contributions of this doctoral study:

6.4.1 Insight into a particular group of interdisciplinary students learning CT:

As different departments attempt to provide their students the opportunity to learn CT, the computer science community is trying to figure out effective and innovative ways to do so. This research focused on an interdisciplinary group of students learning CT. This particular group consists of students from different disciplines, typically non-computer science majors. The study illustrates interaction behaviors of these students and also captures their CT learning experiences with peers in small groups. These characteristics
help us better understand the nature of this particular group of novice CT learners and pave the way for potential future research opportunities.

6.4.2 Qualitative assessment of the cohort model:

The findings of this study suggest that a cohort model (long term student groups) can be considered an effective social practice for helping novice learners overcome the initial hurdle with learning CT. Although students were not required to solve a single problem together, the cohort model still served as a safe space for learners to ask for help, share ideas and socially interact with others.

6.4.3 Adaptation of Chi’s DOLA framework in the context of collaborative CT learning

Social interaction, which is key to any collaborative effort, was recorded, examined, and evaluated to measure learners’ engagement in a face-to-face classroom setup for CT learning (where students were collaborating in small groups) as well as in an online platform (where learners were collaborating in the online community). The outcome of the studies, which adapted the DOLA framework, can be used as an assessment tool to evaluate individual students’ collaborative aspects of learning CT.

6.4.4 Adaptation of Stahl’s framework in the context of collaborative CT learning in a cohort model

The incorporation of Stahl’s “influences of collaborative learning framework” (Stahl, 2010) and Chi’s “Differentiated Overt Learning Activities” (Michelene Chi, 2009) framework was able to illustrate students’ experiences of collaborative CT learning in a cohort model.

6.4.5 Better assess the collaborative nature of a Scratch™ user

Tools that have been designed to assess CT learning (Moreno-León et al., 2015; Ota et al., 2016) of users are primarily focused only on the CT learning aspect. The framework used in the Scratch study (RQ3) enables us to better assess the collaborative nature of a user. The assessment of social interactions can be used to provide guided feedback to the user. For example, a user can be suggested to follow a user with a similar interest in
projects, or suggestive projects (project that have used different types of blocks than the ones the user had been using) for remix can pop up etc.

6.4.6 Research publications and awards

6.4.6.1 The findings of the pilot study have been published in the Proceedings of Innovation and Technology in Computer Science Education (ITiCSE) conference in 2015. The title of the paper is Design and Preliminary Results From a Computational Thinking Course (Kafura et al., 2015)

6.4.6.2 The proposal of this dissertation study was selected in a competitive doctoral consortium in the annual conference of the International Computing Education Research (ICER) in 2015. The title of the proposal abstract was Understanding Collaborative Computational Thinking.

6.4.6.3 The researcher of this dissertation study (along with two other team members) received the prestigious XCaliber Award in 2016 for her contribution in the design, implementation and research conducted on the Computational Thinking course at Virginia Tech. The XCaliber award each year recognizes exceptional, high-caliber contributions to technology-enriched teaching and learning.

6.4.6.4 The initial findings of RQ2 were published as a poster titled Collaborative Interdisciplinary Computational Thinking in the AERA Annual Meeting of 2017 in the session titled Stories From the Field: Integrating Computational Thinking Across Curricular Domains.
CHAPTER 7 : COMPLETE REFERENCE


Kafura, D., Bart, A. C., & Chowdhury, B. (In progress). The Design of a University General Education Course in Computational Thinking.


# APPENDIX A: CT COURSES OFFERED BY DIFFERENT UNIVERSITIES

Table A.1 CT courses offered by different universities (adopted from Kafura, Bart, and Chowdhury (In progress))

<table>
<thead>
<tr>
<th>University</th>
<th>Offered for</th>
<th>Pedagogical approach</th>
<th>Duration</th>
</tr>
</thead>
</table>
| UNC-Charlotte (Senske, 2011) | Architecture students                                                      | • Through digital design projects  
                                 |                                                                                  | • Provide many contexts to foster transfer |
|                          |                                                                            | Duration of courses within the curriculum                                             |
| DePaul University (Perković, Settle, Hwang, & Jones, 2010) | All students (general education courses)                                       | • Solving problems  
                                 |                                                                                  | • Case studies  
                                 |                                                                                  | • A diverse group of faculty were involved in developing a framework |
|                          |                                                                            | Duration is not specified, but computational concepts were explicitly mapped over several courses |
| Carroll University (Kuster, Symms, May, & Hu, 2011) | Students who will do a Bachelor of Science                                    | • Solving problems  
                                 |                                                                                  | • Game design  
                                 |                                                                                  | • Review journal articles in their field |
| Baylor University (Booth, 2013) | All students                                                             | • Collaborative: teams with frequently changing membership work together  
                                 |                                                                                  | • Solving problems  
                                 |                                                                                  | • Interdisciplinary approach: a cross section of faculty were involved in the course design of problems used in the course |
|                          |                                                                            | Duration of one semester course                                                      |
| Purdue University (Hambrusch, Hoffmann, Korb, Haugan, & Hosking, 2009) | Science majors                                                            | • Solving problems                                                                |
|                          |                                                                            | Duration of one semester course                                                      |
| Brown University (Ritz, 2013) | Students of humanities and social sciences                                  | • Analyze printed newspapers by hand  
                                 |                                                                                  | • Problem/Project based: Engaging student interest by using problems of relevance and allowing substantial time for student-defined work |
|                          |                                                                            | Duration of one semester course                                                      |
| Carnegie Mellon University (Cortina, 2007) | Students who will not take another programming course (humanities, programming) | • Solving problems  
                                 |                                                                                  | • Guest lectures involving faculty experts in different topic areas  
<pre><code>                             |                                                                                  | • Term paper exploring social impacts |
</code></pre>
<p>|                          |                                                                            | Duration of one semester course                                                      |</p>
<table>
<thead>
<tr>
<th>Institution</th>
<th>Course Type</th>
<th>Description</th>
<th>Duration</th>
</tr>
</thead>
</table>
| Virginia Tech | Prospective computer science majors | • Provide practical, concrete, learning experiences about a variety of important computing concepts using tools and physical simulation rather than programming
• Solving problems | One semester course |
| National University of Defense Technology | First year students | • Problems from a variety of disciplines to stimulate student interest.
• Interdisciplinary collaborative: students were required to work in teams of 2-3 students which included at least two different majors. | One semester course |
| Tuskegee University | Biology students | • Collaborative: students randomly-assigned to teams | One semester |
| edX | Aimed at students with some prior programming experience and a rudimentary knowledge of computational complexity | • Solving problems | 9 weeks |
APPENDIX B: INFORMAL LEARNING CALCULATIONS BY YANG ET AL.
(2015)

In order to assess learning of CT of group members, Yang et al. (2015)’s model of informal learning has been used. Yang et al. in their model used Inverse Document Frequency (IDF) to assess informal learning in Scratch members. An overview of the IDF is provided next, following which a worked out calculation of learning is shown.

The IDF is a widely used statistical measure used to illustrate how important a word is in a document (Manning et al., 2008; Sparck Jones, 1972). In this regard, a term called Term Frequency (TF) is simply the number of times a word appears in a document. On the other hand, the IDF assigns more weight to key words that appear rarely than the ones that are more commonly used. If the total number of words in a document is \( N \), \( DF_t \) is the number of the times the term \( t \) appears and \( IDF \) of the term \( t \) is as follows:

\[
IDF_t = \log \frac{N}{DF_t} \quad \text{(1)}
\]

(Manning et al., 2008)

IDF has been used by Yang et al. (2015) to illustrate the amount of learning of a scratch user. Yang et al. calculated block weights of the different blocks used by users in their original projects.

\[
w_{b_j} = \log_{10} \frac{1+P}{1+P_{b_j}} \quad \text{(2)}
\]

(Yang et. al, 2015)

Where \( P \) is the total number of original projects, and \( P_{b_j} \) is the number of original projects containing a vocabulary block \( b_j \) (1≤\( j \)≤170). ‘1’ was added to both numerator and denominator to smooth the terms to avoid negative weights and division by zero. Since, Scratch has 170 types of blocks a user can use from, Yang et al.’s IDF weight vector \( W \) has 170 elements, one for each corresponding vocabulary block:

\[
W = [w_{b_1} ... w_{b_j} ... w_{b_{170}}] \quad \text{(3)}
\]
W was used to calculate the vocabulary weights for each block used by certain users in Scratch.

In order to model learning, Yang et al. used the following equations:

1. Align all projects of a user $u$ including original and
2. Construct a $n \times 170$ matrix $P_u$ using the the first $n$ projects and frequency of their 170 blocks:

$$P_u = \begin{bmatrix} f_{1,1} & \cdots & f_{1,j} & \cdots & f_{1,170} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ f_{i,1} & \cdots & f_{i,j} & \cdots & f_{i,170} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ f_{n,1} & \cdots & f_{n,j} & \cdots & f_{n,170} \end{bmatrix} \quad \text{... (4)} \quad \text{(Gelman et al., 2016)}$$

where $f_{i,j}$ is the frequency of block $b_j \leq j \leq 170$ in project $i \leq i \leq n$.

3. Create a matrix $P_c$ by cumulatively summing rows in $P_u$ (e.g., the $i^{th}$ row of $P_c$ is the element-wise sum of the first $i$ rows of $P_u$). The dimension of $P_c$ is $n \times 170$.

4. Create a binary matrix $P_b$ from $P_c$ (‘1’ if frequency of an element $> 0$, ‘0’ otherwise).

5. Compute a trajectory by applying weights on the elements of $P_b$ and summing values in each row: $V_u = (P_b W^T)^T$. The final result is a $n$-dimensional vector:

$$V_u = [v_{u,1} \ v_{u,2} \ \cdots \ v_{u,i} \ \cdots \ v_{u,n}] \quad \text{... (5)}$$

Where $v_{u,i}$ is the cumulative sum of weighted vocabulary blocks for a user $u$, computed using the first $i$ projects. It is weighted binary count considering whether a block was ever used (frequency ignored). $V_u$ can be computed for $M$ number of users to construct a $T_{all}$ that has $M \times n$ dimensions. Each $V_u$ is a row vector of size $n$ (equation 5).

$$T_{all} = \begin{bmatrix} V_1 \\ \vdots \\ V_u \\ \vdots \\ V_M \end{bmatrix} \quad \text{... (6)}$$
Create a vector $O_u$, corresponding to a trajectory that only uses original projects, simply by by replacing rows of remixed projects in matrix $P_u$ with vectors of zeros:

$$O_u = [o_{u,1}, o_{u,2}, \ldots, o_{u,i}, \ldots, o_{u,n}]$$

Compute $T_{ori}$ that has $M \times n$ dimensions

$$T_{ori} = \begin{bmatrix} O_1 \\ \vdots \\ O_u \\ \vdots \\ O_M \end{bmatrix} \quad \ldots(7)$$

The trajectory of vocabulary progress- original & remix, and original can be plotted using corresponding $T_{all}$ and $T_{ori}$ to illustrates the amount of learning progress by various users.

APPENDIX C: WORKED EXAMPLE OF YANG’S (2015) MODEL

Yang’s model of assessing learning has two steps
I. Calculate IDF for each computational block

II. Calculate learning of each user based on IDF values calculated in the first step

Section I provides a worked out example of calculating IDF and Section II provides an example of how learning was calculated for a user in this study.

SECTION I: IDF CALCULATION

**Step 1:** Suppose cluster X has 10 users and in total the 10 users created 100 original projects. IDF will be calculated using these 100 original projects.

Thus, User: 10 and total original project created by all 10 users:100 original projects

**Step 2:** Each project can use 170 different types of blocks as many times as a user needs to. No matter how many times each block has been used in a project, if a user has used a block assign ‘1’ to the block that it used.

<table>
<thead>
<tr>
<th>Project_id</th>
<th>do_until</th>
<th>And</th>
<th>or</th>
<th>doif</th>
<th>not</th>
<th>add</th>
<th>stopallsounds</th>
</tr>
</thead>
<tbody>
<tr>
<td>1920010</td>
<td>5</td>
<td>7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>6</td>
<td>6</td>
</tr>
</tbody>
</table>

Will be turned to

<table>
<thead>
<tr>
<th>Project_id</th>
<th>do_until</th>
<th>and</th>
<th>Or</th>
<th>doif</th>
<th>not</th>
<th>add</th>
<th>...</th>
<th>...</th>
<th>stopallsounds</th>
</tr>
</thead>
<tbody>
<tr>
<td>1920010</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>

**Step 3:** Create a matrix 100X 170 with all binarized blocks of 100 original projects of all users
Step 4: For each block column wise add values. The result will be a 1X170 matrix

<table>
<thead>
<tr>
<th>do_until</th>
<th>and</th>
<th>or</th>
<th>doif</th>
<th>not</th>
<th>add</th>
<th>…</th>
<th>…</th>
<th>stopallsounds</th>
</tr>
</thead>
<tbody>
<tr>
<td>90</td>
<td>88</td>
<td>33</td>
<td>95</td>
<td>25</td>
<td>45</td>
<td></td>
<td></td>
<td>10</td>
</tr>
</tbody>
</table>

In the above matrix the do_until block is calculated to be 90. That means 90 different projects in the 100 projects used the do_until block. Similarly, the add block is calculated to be 45. That means that in 100 projects, 45 projects used the add block.

Step 5: For each block calculated IDF value using the following equation:

\[
w_{bj} = \log_{10} \frac{1 + P}{1 + P_{bj}}
\]

Where P is the total number of original projects, and \(P_{bj}\) is the number of original projects containing a vocabulary block \(b_j\) \((1 \leq j \leq 170)\).

Hence, IDF for the block do_until in our previous step would be

\[
w_{b_{do\until}} = \log_{10} \frac{1 + 100}{1 + 90} = .040
\]
Similarly, IDF vector value for each block is calculated

<table>
<thead>
<tr>
<th>do_until</th>
<th>and</th>
<th>or</th>
<th>doif</th>
<th>not</th>
<th>add</th>
<th>...</th>
<th>...</th>
<th>stopallsounds</th>
</tr>
</thead>
<tbody>
<tr>
<td>.040</td>
<td>.169</td>
<td>.48</td>
<td>.022</td>
<td>.588</td>
<td>.34</td>
<td></td>
<td></td>
<td>.96</td>
</tr>
</tbody>
</table>

**SECTION 2: CALCULATING LEARNING FOR EACH USER**

This section describes how a user’s CT learning was calculated

Step 1: Find users that have at least 5 original projects

Step 2: For each user, select all original projects and the corresponding blocks used in the projects. Suppose, **User A** has **n** original project create a matrix n X 170 which stores the number of times each block has been used in that project.

<table>
<thead>
<tr>
<th>Original projects created by User A and blocks used in the projects</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proj1</td>
</tr>
<tr>
<td>Proj2</td>
</tr>
<tr>
<td>...</td>
</tr>
<tr>
<td>...</td>
</tr>
<tr>
<td>Proj n</td>
</tr>
</tbody>
</table>

**Step 2:** For User A, create another matrix \( P_c \) by cumulatively summing rows in \( P_u \) (e.g., the \( i^{th} \) row of \( P_c \) is the element-wise sum of the first \( i \) rows of \( P_u \)). The dimension of \( P_c \) is n x 170.
Step 3: Create a binary matrix $R_b$ from $P_c$ ('1' if frequency of an element > 0, '0' otherwise)

<table>
<thead>
<tr>
<th>do_until</th>
<th>and</th>
<th>or</th>
<th>doif</th>
<th>not</th>
<th>add</th>
<th>...</th>
<th>...</th>
<th>stopallsounds</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Binary matrix of cumulative blocks used in Original projects created by User A

Step 4: Compute a trajectory by applying weights on the elements of $R_b$ and summing values in each row: $V_u = (P_b W^T)^T$. The final result is a n-dimensional vector:

$$V_u = [v_{u,1} v_{u,2} ... v_{u,i} ... v_{u,n}]$$

Where $v_{u,i}$ is the a cumulative sum of weighted vocabulary blocks for a user $u$, computed using the first $i$ projects. It is weighted binary count considering whether a block was ever used (frequency ignored).
From the first project User A learned \((.626 + .588) = 1.216\)

Cumulative learning of User A from 2 projects is \((.626 + .169 + .022 + .588) = 1.238\)

**Step 5:** The final cumulative vocabulary is the amount learned by a user. In this example User A’s learning amount is 4.147
<table>
<thead>
<tr>
<th>Name of table</th>
<th>Brief description</th>
</tr>
</thead>
<tbody>
<tr>
<td>user table</td>
<td>Contains information about all publicly visible user accounts (user_id, country, date_created)</td>
</tr>
<tr>
<td>projects table</td>
<td>Contains metadata about publicly available projects and their creators( user_id, project_id, views, lover,downloads, block types, is_remix, parent_user_id, parent_project_id etc)</td>
</tr>
<tr>
<td>galleries table</td>
<td>Contains metadata about all publicly available galleries. A gallery is a web page that presents a collection of projects. Galleries are created by a single user</td>
</tr>
<tr>
<td>friends table</td>
<td>Contains the unidirectional relationships between users, displayed in the website as “friends.”</td>
</tr>
<tr>
<td>downloaders table</td>
<td>Includes each unique download of a project by a user (project_id, date downloaded)</td>
</tr>
<tr>
<td>favoriters table</td>
<td>Includes each “favorite” given to publicly available projects. A user can favorite a project by clicking on the “add to favorites?” link in a project’s page</td>
</tr>
<tr>
<td>lovers table</td>
<td>Represents the clicks of the heart-shaped &quot;love-it&quot; button that appears on every project page. This action is socially framed as a positive expression of appreciation for a project (project_id, date_loved)</td>
</tr>
<tr>
<td>projects_text table</td>
<td>Contains the free-form and the unstructured text fields in the projects table (project_id, title, description)</td>
</tr>
<tr>
<td>galleries_text table</td>
<td>contains the free-form and the unstructured text fields in the galleries table (gallery_id, title, description)</td>
</tr>
<tr>
<td>Pcomments table</td>
<td>Contains the metadata about comments posted on projects. This dataset contains a row for each comment and metadata on who, when, and in which project, each comment was posted.</td>
</tr>
<tr>
<td>pcomments_text table</td>
<td>Contains the free-form and the unstructured text fields in the pcomments table (pcomment_id, text)</td>
</tr>
<tr>
<td>gcomments table</td>
<td>Contains the metadata about comments posted on galleries. This dataset contains a row for each comment that includes who, when, and in which gallery each comment was posted.</td>
</tr>
<tr>
<td>gcomments_text table</td>
<td>Contains the free-form and the unstructured text fields in the gcomments table (gcomment_id, text)</td>
</tr>
</tbody>
</table>