Supporting the Integrated Learning of Science, Engineering, and Computational Thinking in an Open-ended Learning Environment

By

Ningyu Zhang

Dissertation

Submitted to the Faculty of the

Graduate School of Vanderbilt University

in partial fulfillment of the requirements

for the degree of

DOCTOR OF PHILOSOPHY

in

Computer Science

August 7, 2020

Nashville, Tennessee

Approved:

Gautam Biswas, Ph.D.
Corey Brady, Ph.D.
Douglas Fisher, Ph.D.
Jennifer Chiu, Ph.D.
Maithilee Kunda, Ph.D.

ACKNOWLEDGMENTS

First and foremost, I would like to thank my advisor and dissertation chair Professor Gautam Biswas. I was fortunate to be advised and supported by Dr. Biswas since I joined the Teachable Agents / Open-ended Learning Environment Research Group in 2014. Dr. Biswas set a role model for me for his tireless pursuit of understanding how people learn. I also want to thank Professors Corey Brady, Douglas Fisher, Jennifer Chiu, and Maithilee Kunda for guiding me through my coursework and serving on my dissertation committee.

Thanks to my wonderful wife Siruo Li, my amazing daughter Shuwei Zhang, and our families for their support and companion. My two-year-old's insistence on sorting everything by the colors of the rainbow sparked the idea of coding the data collected for the study in red, orange, yellow, green, and blue.

Thanks to the SPICE project co-PIs Kevin McElhaney and Satabdi Basu who inspired ideas and supported my research. Thanks to the collaborators from the University of Virginia and SRI International, including but not limited to Sarah Fick, Ron Fried, Reina Fujii, and Nonye Alozie. I would also like to thank the classroom teachers Ms. J.S., Mr. J.P., and the cohorts of students with whom I work since 2015 on various research projects.

Last but not least, to my friends, fellow graduate students and lab members Shunxing Bao, Fangzhou Sun, John Kinnebrew, Jim Segedy, Cheng Ye, Yi Dong, Brian Broll, Brian Gauch, Emily Darwin, Gerald Stanje, Michael Tscholl, Mona Emara, Ramkumar Rajendran, Naveed Mohammed, Anabil Munshi, Nicole Hutchins, Shitanshu Mishra, Caitlin Snyder, Ben Yett, and Allison Moore. I shared a very important time of my life with you and learned so much from you.

This material is based upon work supported by the National Science Foundation under Grant No. DRL-1742195 and Grant No. Cyberlearning 1441542. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

This dissertation adopts a systematic computational thinking (CT) framework to develop computer-based learning environments (CBLEs) for the integrated **science** and **engineering** learning in upper elementary and lower middle school classrooms. More specifically, CT concepts and practices are instantiated through computational modeling activities in a **learning-by-modeling** environment where students build computational models of science phenomena and then use the developed models to solve an engineering design task. A series of summative and formative assessments were designed to assess student learning. We also developed innovative schemes to analyze students' strategic learning behaviors as they perform their computational model building and engineering design tasks.

1.1 Science, Engineering, and Computational Thinking in K-12 Education

The majority of people in the United States learn most about science and engineering in middle and high schools, and the school-based experience during these formative years deeply shapes their perspectives and future interactions with science and engineering (Pianta et al., 2007; Maltese and Tai, 2010).

On the other hand, summarizing over 60 years of research on students' perceptions of science, Wyss et al. (2012, p. 503, italics added) revealed that "*students do not have a clear perception of what science has to offer them or what scientists do*." Similarly, a large proportion of students have limited understanding of engineering, with some still perceiving engineers as manual-labor workers assembling vehicles or doing construction work (Fralick et al., 2009; Cunningham, 2017). These misunderstandings further exacerbate the loss of interest in science, technology, engineering, and math (STEM) subjects (Sadler et al., 2012; George, 2006), and have a long-term impact because they affect future occupation choices in STEM fields (NAE and NASEM, 2019). Therefore, it is imperative to introduce appropriate science and engineering curricula in the early stages of K-12 education.

On another thread, Computational Thinking (CT) has been recognized as a framework for developing computer literacy and computing skills among the K-12 computer science (CS) and STEM communities (Barr and Stephenson, 2011; Grover and Pea, 2013; Sengupta et al., 2013; Weintrop et al., 2016a; Shute et al., 2017). The industry also considers CT to be one of the primary drivers of the 21st-century workforce (Barr et al., 2011). In general, CT encompasses a wide range of abilities and practices, such as problem decomposition, algorithm development, reasoning at multiple levels of abstraction, and creating and reusing modular solutions (Wing, 2006, 2011; Shute et al., 2017).

Researchers and practitioners believe that by drawing from the fundamental skills and practices of CT, students can develop analytical and problem-solving skills and practices. These CT practices go beyond learning how to write code in CS, benefiting students' understanding of scientific processes, engineering design, and human behaviors (Wing, 2006; NRC, 2010; Barr and Stephenson, 2011; NRC, 2011). In other words, CT can foster K-12 students' learning in STEM domains as they solve problems while thinking like computer scientists (Barr and Stephenson, 2011; Wing, 2011).

The acknowledgment of these benefits of CT has promoted thinking about including CT into the K-12 STEM curricula; for example, the Next Generation Science Standards (NGSS) in the United States include CT as a core scientific practice (NGSS Lead States, 2013; Barr and Stephenson, 2011). Researchers have also stressed the urgency of introducing CT into K-12 classrooms. However, there has been insufficient effort to include CT into the disciplinary curricula, especially those integrating science and engineering.

1.2 Artificial Intelligence in Education and Computer-based Learning Environments

The 21st century has witnessed considerable advances in artificial intelligence in education (AIED). AI and machine learning methods are widely used by educators and learners today (Stone et al., 2016). It is believed that intelligent learning systems will become an essential part of the teaching and learning processes in K-12 and undergraduate classrooms. In the next 15 years, human teachers will be assisted by AI technologies both in the formal classroom settings and informal settings outside the classroom, as its application will facilitate more customizable approaches to learning (Stone et al., 2016).

Successful implementation of AIED often involves the use of computer-based learning environments (CBLEs). An important component of the success in CBLEs is rooted in the data collected as the learners engage in learning activities. Data sets collected from CBLEs have fueled the rapid growth of the field of learning analytics and data mining (Siemens and Baker, 2012) that employ a wide range of AI techniques, such as supervised and unsupervised learning methods, natural language processing, and deep learning to analyze key learning constructs, such as students' engagement and learning outcomes (Stone et al., 2016), and learners' cognitive and metacognitive processes (e.g., Baker et al. (2010); Munshi et al. (2018); Segedy et al. (2015b); Kinnebrew et al. (2017); Basu et al. (2017)). In addition, aggregated data has been used to detect students' common misconceptions (e.g., Chi et al. (2012)), predict the risk of failure (e.g., Cortez and Silva (2008)), and provide real-time student feedback that is linked to the learning outcomes and behaviors (e.g., Basu et al. (2017)).

1.3 Scope and Contributions of this Dissertation Research

This dissertation is built upon a systematic CT-based framework for integrated learning of science and engineering as students engage in activities for constructing, testing, and refining computational models and using these models to generate, compare, and revise engineering designs. It hypothesizes that the integration of these disciplines will lead to instructional and learning benefits for the teachers and students. Specifically, the hypotheses investigated under this dissertation are:

1. Science learning and engineering design share a synergistic relationship, i.e., the mastery of one construct facilitates the other.

2. Computational thinking can play an important enabling and integrating role in the learning of both domains (science and engineering).

3. Computational modeling and engineering design tasks are complex and require the use of multiple cognitive processes for successful execution. These complex tasks also require self-monitoring and strategic thinking to achieve successful outcomes. Therefore, it is imperative to study and support novice learners in developing appropriate cognitive and metacognitive strategies to accomplish their learning and problem-solving goals.

To accomplish these goals, this dissertation makes additional contributions to developing methodologies to address:

1. The lack of research investigating the integration of science and engineering facilitated by CT (Irgens et al., 2020; Magana and de Jong, 2018).

2. The complexities of automated evaluation and scoring of student-generated solutions (computational modeling and engineering design) in open-ended learning environments (OELEs) (Land, 2000).

3. The insufficient understanding of the affordances and benefits of an integrated curriculum that brings together science, engineering, and CT (Zhang and Biswas, 2019).

4. The need to develop methods to evaluate and link students' learning and problem-solving processes using state of the art analytic and mining methods (Zhang et al., nd).

1.4 Organization of this Thesis

This dissertation is organized as follows. Chapter 2 provides the **background and literature review** covering four related areas of (1) science learning and engineering in K-12 settings, especially in the lower grades, (2) Computational Thinking (CT), (3) learning by modeling and open-ended Learning environments (OELEs), and (4) learning strategies. The previous work conducted in these areas motivates the research presented and the contributions made in this dissertation.

Chapter 3 provides a formal **statement of the problems** addressed in this dissertation. It revisits the critical summary of the literature review and presents the research questions and research hypotheses introduced in Chapter 1.3.

Chapter 4 introduces **the learning environment**, focusing on adopting an evidence-centered design (Mislevy et al., 2003) approach to (1) unpacking of the domain concepts and practices used in the curriculum, (2) translating the curriculum requirements to designing and implementing an OELE that guides and supports student learning, (3) the software infrastructure enabling logging of relevant student work (models and engineering designs) and the actions used to generate the artifacts, and (4) the methods and techniques for evaluating and understanding students' learning performance and behavioral processes in the environment.

Chapter 5 presents the **methods** used in the dissertation, including (1) a description of a classroom study with 99 6th-grade students in the middle Tennessee region, (2) the formative and summative assessments used in this study, (3) analyses of the log data collected from the students in the study using the computational modeling and engineering design environment presented in Chapter 4, and (4) the quantitative analytical techniques and machine learning techniques to analyze and understand student learning.

Chapter 6 presents and discusses the **results** derived from data collected in the classroom study focusing on the students' learning outcomes in the summative assessments, formative assessments, computational model-building behavior and performance, and engineering design behavior and performance. Statistical inference methods, such as hypothesis testing, analysis of variance (ANOVA), analysis of covariance (ANCOVA), and correlation analysis (Kokoska and Zwillinger, 2000; Lilliefors, 1967; Lowry, 2014; Siegal, 1956) are applied to investigate the relations between these learning outcomes. Finally, a Path Analysis (Wright, 1983) was applied to model the relations between the different outcome variables derived in the study.

Chapter 7 examines and analyzes students' use of **learning strategies** in the OELE. It uses the concepts developed in Chapters 2 and 3 and approaches presented in Chapter 5 to model, detect, and evaluate learning strategies and link these strategic learning behaviors to the learning outcomes discussed in Chapter 6.

Finally, Chapter 8 discusses the contributions of this dissertation and outlines future work to advance the present research.

Chapter 2

Background and Literature Review

This chapter serves as the background and reviews the four components that support the conceptual framing for this dissertation:

- Science and engineering learning in K-12 settings,
- Computational Thinking (CT) as the platform for supporting integrated science and engineering learning,
- learning-by-modeling and problem-solving in open-ended Learning environments (OELEs), and
- learning strategies as the basis for analyzing students' approaches to their learning and problem-solving tasks;

This chapter provides a brief overview of the state-of-the-art in these areas and then establishes the contributions of this dissertation in relation to the state of the art.

Section 2.1 provides a review of the integration of science and engineering in K-12 education drawn from national standards and committee reports such as National Research Council (2012); NGSS Lead States (2013), and NAE and NASEM (2019). In addition, Section 2.2 reviews and critiques a few prominent examples linked to the integration of science and engineering learning. These include Design Human Elbows (Penner et al., 1998), the LEGO^{TM} design challenges (Wendell and Rogers, 2013), and Learning by Design^{TM} (Kolodner et al., 2003a).

Section 2.3 provides a review of CT, with an emphasis on the definitions and key components of CT that can be linked to STEM learning (e.g., Sengupta et al. (2013); Weintrop et al. (2016a); Basu et al. (2017)). Section 2.3.2 summarizes the affordances of learning and applying CT from recent research projects and meta-analyses.

Section 2.4 reviews the affordances and challenges that students face with *learning-by-modeling* approaches, especially those involving computational modeling activities. It also introduces open-ended learning environments (OELEs) —a specific form of computer-based learning environments that incorporate AI methods to support and scaffold learning-by-modeling approaches.

Finally, Section 2.5 reviews related work in learning strategies –a topic that is closely tied with helping novice learners become more successful and effective in their learning and problem-solving tasks. Section 2.5.1 reviews the prominent strategy frameworks linked to OELEs.

2.1 Engineering Design and Science Learning in K-12 Education

Design is a core component of engineering (Cunningham et al., 2007; National Research Council, 2012). It invokes cognitive processes such as (1) understanding the problem, (2) generating ideas, (3) learning new concepts necessary for solving problems, (4) developing and testing models, and (5) analyzing and revising solutions to accomplish problem-solving goals (Mehalik et al., 2008).

In the professional world, while conducting investigations, scientists and engineers "*determine what needs to be measured; observe phenomena; plan experiments, and methods of data collection; build instruments; engage in disciplined fieldwork; and identify sources of uncertainty*" (National Research Council, 2012, p. 45, italics added). Engineers also frequently employ modeling and simulation practices to assist innovation with experimentation and theoretical approaches (Magana and de Jong, 2018).

In the K-12 setting, students can identify themselves with practicing and engaging in science and engineering activities in substantive and meaningful ways, such as asking productive questions, analyzing the problem, and generating solutions (Hirsch et al., 2007). K-12 students can view engineers as creative, future-oriented, and artistic problem finders and solvers even though they have little experience with complex engineering practices (NAE and NASEM, 2019; English et al., 2011).

Following the ideas outlined in *A framework for K-12 science education: Practices, crosscutting concepts, and core ideas* (NGSS Lead States, 2013) (referred to as the *Framework* below), the core idea of engineering design includes the following three components in the K-12 context:

- *Defining and delimiting engineering problems involves stating the problem to be solved as clearly as possible in terms of criteria for success and constraints or limits*;
- *Designing solutions to engineering problems begins with generating a number of different possible solutions, then evaluating potential solutions to see which ones best meet the criteria and constraints of the problem*;
- *Optimizing the design solution involves a process in which solutions are systematically tested and refined and the final design is improved by trading off less important features for those that are more important*; (NGSS Lead States, 2013, App. I, italics added)

Scientific investigation and engineering design activities are more effective for supporting learning by engaging students in learning as well as by increasing their conceptual knowledge, reasoning, and problem-solving skills (NRC, 2000; NAE and NASEM, 2019). As stated in a report published by the U.S. National Academy of Engineering and National Academies of Sciences, Engineering, and Medicine, the K-6 science curricula should focus on "*Engaging students in learning about natural phenomena and engineering challenges via science investigation and engineering design increases their understanding of how the world works*" (NAE and NASEM, 2019, p. 4, italics added).

Whereas science learning through problem-solving has received a lot of attention in secondary and post-secondary curricula, e.g., (Kolodner et al., 2003a; Magana and de Jong, 2018), there is much less focus on *science-through-design* learning for younger pupils (Wendell and Rogers, 2013). Magana and de Jong (2018) listed a few initiatives proposed by education stakeholders such as the American Society for Engineering Education (ASEE) and the Accreditation Board for Engineering and Technology (ABET) to engage students with professional science and engineering practices. On the other hand, curricula that were promoted with the incorporated learning focused on the undergraduate engineering curriculum.

Engineering has not traditionally been part of the K-12 curriculum. Instead, it is often taught as a stand-alone elective course, where students primarily work on design projects with little discussion of the science that supports the design and implementation (Cunningham et al., 2007). In addition, due to the limits of integrated curricula, schools and teachers often have to take the extra step to implement their own curriculum (Moore et al., 2015).

As a response, there has been "*growing inclusion of engineering design in K-12 classrooms*" that holds great potential and opportunities for all students to construct an understanding of the natural and designed world and develop an agency (NAE and NASEM, 2019, p. vii, italics added). Rethinking the roles of engineering and experimentation gives rise to the trend of inclusion. For example, National Research Council (2006) documented a shift away from viewing experiment and design experiences as separate from classroom science instruction and suggested that integrated experiments were essential steps to drive towards the integrated teaching and learning exercises: "*Laboratory experiences provide opportunities for students to interact directly with the material world (or with data drawn from the material world), using the tools, data collection techniques, models and theories of science*" (National Research Council, 2006, p. 31, italics added).

There are also changes in the conceptualization of the nature of learning —the learning of science content and the method of doing science are considered to engage simultaneously within the three dimensions of learning: *science and engineering practices, disciplinary core ideas*, and *crosscutting concepts* (National Research Council, 2012; NAE and NASEM, 2019). One of the important conclusions and guidelines from the *Framework* is that "*instead of seeing engineering as separate from science, students can see the ways science and engineering each serve the other*" (NAE and NASEM, 2019, p. 2, italics added).

Building upon these conceptual changes, the *Framework* proposed that *science investigation* and *engineering design* should be more central to curricula that include students' investigation of scientific phenomena and designing solutions to make sense of the cause and solve the challenges of interest (NAE and NASEM, 2019). Engineering design activities are endorsed as "*to the same level as scientific inquiry in science classroom instruction at all levels, and by emphasizing the core ideas of engineering design and technology applications*" (NGSS Lead States, 2013, Executive Summary, p. 1, italics added).

In addition, the integrated learning should be supported by curricula built around students' investigation of scientific phenomena and designing solutions to make sense of the cause or solve the challenges of interest because learning is meaningful when the design and investigation activities are relevant to student lives (NAE and NASEM, 2019; Magana and de Jong, 2018; Xing et al., 2019). On the practical side, both teachers and students could understand the engineering design process even for students who had not been successful in traditional science courses (Cunningham et al., 2007). In addition, the engineering pedagogy has great potential to increase students' achievement and interest in STEM disciplines (Moore et al., 2015).

Science classes provide good contexts for practicing engineering activities and receiving benefits from them. Some existing studies indicate that the integration of modeling and simulation improved the conceptual learning of undergraduate engineering students (Magana and de Jong, 2018). When students participate in learning activities such as investigating scientific scenarios and making engineering designs, they effectively engage with the science and engineering phenomena being investigated as they make questions, collect and analyze data, organize evidence, and formulate models to support reasoning and justify solutions (NAE and NASEM, 2019). Such deep engagement can lead to stronger conceptual and understandings of science content as opposed to traditional memorization-centered approaches.

In addition, a design-based engineering instruction is more accessible to younger learners as they tend to have “*less apprehension toward design challenges*" compared to older learners (Wendell and Rogers, 2013, p. 515, italics added). In addition, engineering design activities have great potential and promise to benefit science learning because the scientific scenarios can be contextualized into compelling design problems (Kolodner et al., 2003a; Kafai and Ching, 2001; Wendell and Rogers, 2013). These call for the integration of science learning and engineering education at the K-12 level, especially early stages.

2.2 Related Work of Design-based Learning

This section provides a review and critique of the representative projects and related work that supported engineering design activities.

2.2.1 The Spring and Water Canal Tasks

In their work on investigating how different modalities of thinking during experimentation influenced middle school students’ learning of physics, Schauble et al. (1991) found students facing challenges such as difficulties in drawing causal relationships between observed variables. For example, when mixing chemical substances, instead of performing scientific investigations, young learners might focus on manipulating the variables to produce a desirable outcome, such as getting a particular color they like. Schauble et al. (1991) described this type of behavior as an *engineering model*, which is a practical approach to experiment to optimize an outcome. In contrast, the investigation of the causal relations among variables is defined as the *science model* (Schauble et al., 1991). The science model emphasizes understanding the causal relations and evaluating the evidence before reaching a conclusion; on the other hand, learners adopting the engineering model tend to explore the experiment space partially and make invalid judgments (Klahr and Dunbar, 1988). In addition, the learner might transition from the engineering model to the science model as they switch the goal from generating the desired outcome to understanding the relevant principles and rules (Karmiloff-Smith and Inhelder, 1974; Schauble et al., 1991)

Schauble et al. (1991)'s dichotomy of learner's exploratory behaviors provided insights into different modes of thinking in an investigative process as the engineering model and the science model. However, this may be an oversimplification since it did not investigate the synergy between the two types of learning modes. For example, this work did not link the two activities together nor investigate how they influenced each other.

2.2.2 Designing Human Elbows

Penner et al. (1998) reported third-grade students involved in designing the natural mechanical systems of the human elbow. The complex task of constructing a deep understanding of the system was made feasible for young learners through a process of iterative modeling (Penner et al., 1997, 1998). The students engaged in a series of design-related activities such as building, testing, and evaluating their design of the human elbow models. They then used the elbow models to explore the biomechanics of the human body (Penner et al., 1998). One of the findings of this study revealed that young, novice learners were able to transition from concentrating on the superficial, *structural* similarity of their designed elbow (e.g., the appearance of the body structure) to the *functional* properties of the mechanical system model (e.g., the motion of the joint) (Penner et al., 1998).

2.2.3 The LEGO^{TM} Design Challenge

Constructions with LEGO^{TM} bricks can create complex and reliable engineering products while remaining an open-ended learning experience for young learners. On the other hand, despite its popularity, research on how it impacts student learning was not well-established (Brophy et al., 2008). Wendell and Rogers (2013) implemented a LEGO^{TM} design challenge for elementary-grade students that created the synergy between science learning knowledge and engineering design. They investigated whether an engineering design-based curriculum had an impact on elementary school (grades 3-5) students' science knowledge and their attitude towards science. Their study found that science content performance significantly improved in comparison to a control group that did not participate in the LEGO$^{\text{TM}}$ Challenge. On the other hand, Wendell and Rogers (2013) also reported minimal differences in the attitudes towards science for the two student groups that did/did not work on the challenge.

2.2.4 Learning by Design$^{\text{TM}}$

Learning by Design$^{\text{TM}}$ (LBD) is a project-based inquiry approach to learning science fortified by case-based reasoning and problem-solving (Kolodner et al., 1998, 2003a,b). The goal of LBD was to support learners to become successful thinkers and decision-makers in the modern world (Kolodner et al., 2003b). LBD supports deep science learning in an environment where students can practice science concepts and skills in parallel with social and communication skills (Kolodner et al., 2003b). Students learn science contents through "*achieving design-and-build challenges*" (Kolodner et al., 2003b, p. 496, italics added). LBD learning activities provide sequenced design-based learning tasks of instantiating science and engineering practices (Kolodner et al., 2003b).

2.2.5 Wearing the Web

The *Wearing the Web* project is a Maker activity cf. (Blikstein, 2013) in a year-long computing course implemented in the Computational Thinking for Girls (CT4G) project (Brady et al., 2016). CT4G adopts a theoretical framework that promotes students' engagement in CT and engineering design (Making) through social computing and simulation activities (Brady et al., 2016). In the Wearing the Web activity, students programmed hardware badges while exploring fundamental CT concepts and practices including network, abstraction, and data representation (Brady et al., 2016). A pilot study exclusively involving girl participants from traditionally underrepresented groups in STEM reported the project's positive effects on students' attitudes towards and interest in computer science.

2.2.6 Summary

The projects reviewed in Section 2.2 have trailblazers in promoting the integration of science learning and engineering education. On the other hand, they all tend to focus on the construction of physical, tangible artifacts. As a result, the affordances of CT (Wing, 2011; Shute et al., 2017) (reviewed in the following section) are not fully utilized.

2.3 Computational Thinking

As early as 1975, computation was suggested as a key method to conduct scientific research by Nobel Laureate Ken Wilson, who argued that computation had become *the third leg of science*, along with theory and experimentation (Denning, 2009). Computational Thinking (CT) has been recognized as a framework for developing computer literacy and computing skills among the K-12 computer science (CS) and STEM (Science, Technology, Engineering, and Mathematics) communities (Barr and Stephenson, 2011; Grover and Pea, 2013; Sengupta et al., 2013; Weintrop et al., 2016a; Shute et al., 2017; Irgens et al., 2020). CT encompasses a wide range of abilities and practices, such as problem decomposition and composition, algorithm development, reasoning at multiple levels of abstraction, and creating and reusing modular solutions (Wing, 2006, 2011; Shute et al., 2017).

2.3.1 Definitions of CT

A seminal work in the CT literature described CT as "solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science" (Wing, 2006, p. 33). Another popular definition of CT states that it is a problem-solving process that includes characteristics such as problem formulation, data analysis, abstraction, algorithmic automation, solution refinement, and transfer (ISTE and CSTA, 2016). Given the wide scope of CT, there are different ideas of what constitutes CT, and how to integrate CT into the K-12 curricula, especially for STEM topics (NRC, 2010; Brennan and Resnick, 2012). In addition, the close relationships between CT, mathematics, algorithmic thinking, and problem-solving can sometimes veil the core ideas in computation that CT encompasses (Rees et al., 2016; Weintrop et al., 2016a).

Some effort has been made to establish operationalizable frameworks for CT. These frameworks either focus on the constructs that emerge from the existing game-based or media-narrative programming environments, or they emphasize STEM concepts and practices. The CT frameworks associated with AgentSheets (Repenning et al., 2000) and Scratch (Brennan and Resnick, 2012) are examples of the former type, and the CT taxonomies proposed by Weintrop et al. (2016a) are examples of the latter type. The CT framework in AgentSheets focuses on the aspects of CT that can be transferred to other learning contexts (Basawapatna et al., 2011). In their framework, CT is defined as the recurring program patterns that students acquire while programming games and later reuse in other simulation contexts (Basawapatna et al., 2011). Brennan and Resnick (2012) defined CT as a framework of three components related to programming in Scratch: what students should know about programming, how to program, and the socio-cultural aspects of programming. More specifically, computational concepts of their framework refer to the fundamental knowledge of computing, such as how loops and conditionals work in a Scratch program. Computational practices are defined as programming related actions, such as building and debugging. Finally, computational perspectives describe learner's computation-related world-view (Brennan and Resnick, 2012).

These frameworks operationalize programming-centered CT constructs in existing environments, but they do not provide explicit evidence of how CT is linked to STEM learning. On the other hand, the CT taxonomies proposed in Weintrop et al. (2016a) emphasize the application of CT in STEM classrooms. Weintrop et al. (2016a) proposed key CT practices that are naturally linked to STEM concepts and practices (NGSS Lead States, 2013), including (1) data, (2) modeling and simulation, (3) problem-solving, and (4) systems thinking. In addition to focusing on what students learn about CT, these CT practices define how students can learn and apply CT, thus providing a theoretical foundation for integrating CT in STEM classrooms.

Some CT frameworks also include the assessment of CT. For example, computational thinking pattern analysis (CTPA) in AgentSheets matches the recurring program patterns in game design and science simulation contexts to evaluate students' understanding of CT (Basawapatna et al., 2011). Additionally, Scratch uses multi-modal assessments including project analyses and interviews to assess its main CT constructs (the CT concepts, practices, and perspectives) (Brennan and Resnick, 2012). These CT frameworks provide important information on students' understanding of CT, but they have their shortcomings as well. For example, students' expression of CT is demonstrated at the program level, assuming that the student understands CT if they use a CT pattern. On the other hand, the used = learned assumption poses peril because a student writing correct programs nevertheless may not have made the necessary conceptual connections (NRC, 2010). In addition, these analyses of snapshots of completed programs lose the temporal information and the subtlety to understand students' developmental process of CT. As a result, many fundamental aspects of CT have not received sufficient attention especially in the context of block-based programming environments, except for a few successful assessments (e.g., Grover and Pea (2013)). Therefore, more detailed, reliable, and formative test instruments need to be developed to enrich CT assessments.

2.3.2 Affordances of CT

Despite the different opinions of the definition and constitution of CT, the research community and educational stakeholders acknowledge that the CT concepts and practices can benefit learning (Zhang and Biswas, 2019). Researchers and practitioners believe that by drawing from the fundamental skills and practices of CT, students can develop analytical and problem-solving skills and practices. These CT practices go beyond the learning of CS and benefit students' understanding of scientific processes, engineering design, and human behaviors (Wing, 2006; NRC, 2010; Barr and Stephenson, 2011; NRC, 2011). In other words, CT can benefit K-12 students' learning in other domains such as mathematics and science as they solve problems while thinking like a computer scientist (Barr and Stephenson, 2011; Wing, 2011).

A series of studies have shown that applying CT in STEM domains helps students' learning (Basu and Biswas, 2016; Basu et al., 2017; Rees et al., 2016; Weintrop et al., 2016a). Additionally, CT skills and practices can transfer to other learning and problem-solving contexts (Basawapatna et al., 2011; Grover et al., 2015), as CT requires a deep understanding of problem-solving when compared against rote learning (Wing, 2006). Therefore, it provides an essential framework for preparing students for future learning (NRC, 2000). In addition, CT includes the practice of designing artifacts (e.g., building models of STEM phenomena), which helps students develop perspectives of the world around them, e.g., a deeper understanding of the role of vegetation in reducing greenhouse gases (Brennan and Resnick, 2012). Therefore, CT provides a synergistic framework for the learning of computational and science concepts and practices (Sengupta et al., 2013; Basu et al., 2017).

Studies have also shown that CT and STEM subjects shared a reciprocal relationship. There is evidence in the literature that students improved their understanding of STEM topics when they are studied in a CT framework (e.g., (Basu et al., 2017; Sengupta et al., 2013; Weintrop et al., 2016a,b; Hutchins et al., 2020). Similarly, developing CT concepts and practices in a science learning framework provides a context and a perspective for a better understanding of CT. For example, the NRC (2010) report states that CT concepts and practices are best acquired when studying them within domain disciplines. If students were introduced to CT in programming contexts only, they might not develop the skills to apply the generalized CT concepts across disciplines because of the difficulties in the transfer of learning (NRC, 2011). Additionally, learning CT concepts and practices in a STEM modeling and simulation framework provides students with a real-world perspective

(Brennan and Resnick, 2012) they may need to develop a good understanding of the STEM and CT concepts in context.

2.4 Learning by Modeling and Open-ended Learning Environments

The construction of models, both physical and computational, has become a prominent way to learn STEM subjects (National Research Council, 2012; Schwarz et al., 2009). Modeling activities, alongside with mathematical analysis, are effective and essential ways to support engineering design (Penner, 2000; National Research Council, 2009; Hmelo et al., 2000). A scientific model is defined as an abstract and simplified representation of a system of phenomena built around the key features needed to explain and predict the scientific phenomena. (Harrison and Treagust, 2000; Schwarz et al., 2009; Gilbert et al., 1998). The generative nature of models has made them available to predict natural phenomena and explains why the system behaves in the way it does (Schwarz et al., 2009).

In a model representation, the *elements, relations, operations*, and *rules* that define the interactions can be expressed using an external notation system in a way that mimics the physical world representation (Lesh et al., 2003; Gilbert et al., 1998). In addition, a constructed model should not only include the representations of the phenomenon and its underlying mechanisms but also be useful for testing hypotheses and answering questions (Gilbert et al., 1998).

The *learning-by-modeling* approach is fundamentally rooted in Papert's theory of constructionism (Papert, 1986; Papert and Harel, 1991), where learning is defined as the construction of knowledge in the socio-cultural context of building tangible objects as artifacts. Within the learning-by-modeling framework, students engage in authentic and self-regulated learning by acting on the materials and structures in the learning environment (Louca and Zacharia, 2015).

2.4.1 Affordances of the Learning-by-Modeling Approach

Affordances of computational modeling activities that support the learning of science and engineering in a virtual environment include (1) enabling learners to conduct experiments on unobservable phenomena, and (2) improving the efficiency and reducing unanticipated consequences of experimental studies (de Jong et al., 2013). For example, invisible or long-term processes such as chemical reactions or geological changes can be manipulated as simulations with computational models. As a result, learners have more opportunities to conduct experiments and gather more information, as compared to conducting observations in a physical environment (de Jong et al., 2013).

Learning-by-modeling curricula have been more effective when compared against traditional approaches: lower-grade students who learned with models outperformed high school students on resolving real-world-situated physics problems using Newtonian mechanics (NRC, 2000). Educators and cognitive scientists have suggested that the systematic building of computational models may help students gain a deep understanding of scientific phenomena (NRC, 2010; Wilensky and Reisman, 2006). A systematic modeling process includes problem decomposition, exploration, and understanding of the relationships between components. Modeling activities can help students develop an understanding of a wide range of scientific constructs (NRC, 2000; Land, 2000). Students' engagement in modeling activities supports the construction of scientific expertise and epistemic understanding (Lehrer and Schauble, 2006; Schwarz and White, 2005).

Learning-by-modeling approaches also lead to instructional benefits of improved domain knowledge and model construction skills (VanLehn, 2013; Chin et al., 2010; Louca and Zacharia, 2015). For instance, modeling supports a more effective conceptual understanding of the nature of science (Harrison and Treagust, 1996). Finally, it is argued that the core elements of scientific modeling practices can be performed and mastered by even elementary school students as they succeed in making model-based predictions as well as evaluating, comparing, and revising models (Schwarz and White, 2005; Lehrer and Schauble, 2006; Schwarz et al., 2009)

Science education standards have also recognized the benefit and importance of teaching students about systems thinking (Cheng et al., 2010; NRC, 2000) and CT (Sengupta et al., 2013; Shute et al., 2017). On the other hand, the field of learning sciences has lacked effective and efficient methods to teach model construction (VanLehn, 2013). Louca and Zacharia (2015) examined the literature and pointed out the scarcity of related research on learning-by-modeling activities (modeling-based learning in their term) for K-6 novice learners. There was also little information known about how young novice learners differ from elder learners and experts (Louca and Zacharia, 2015).

2.4.2 Challenges of the Learning-by-Modeling Approach

It is often a challenging task for novice learners to construct models. Model construction is typically a more complex process than model exploration. Exploration with a model may not require knowledge of the modeling language or involve the complex cognitive processes of problem-solving, such as formulating questions, computing numerical outcomes, interpreting results, and validating solutions (VanLehn, 2013). On a deeper level, learners face many challenges in model construction. As summarized by VanLehn (2013), these challenges are often associated with

- difficulties in understanding the modeling language, for example,
  - losing track of symbols and notations (VanLehn, 2013)
  - having hardship with the idea of *stock* and *flows* of the system dynamics (Hopper and Stave, 2008)
- difficulties understanding the presentation system (Koedinger et al., 2008)
- poor problem-solving strategies, for example,
  - failing to decompose the system into parts (Hogan and Thomas, 2001)
  - not being attentive to the model prediction (Metcalf et al., 2000)

- performing shallow comparisons with the simulation output (Löhner et al., 2005)

More specifically, young learners are characterized as focusing on the representation of the mechanism and ignoring the phenomenon and causal relations of the system under study (Sins et al., 2005; Penner et al., 1998) as they tend to view a modeling task *only* as an engineering problem that requires an optimized solution rather than the scientific reasoning (Sins et al., 2005) As discussed earlier in Section 2.2 concerning the integration of science and engineering learning, a similar, persistent challenge was also documented by Schauble et al. (1991) in their discussion of the science model (understanding the causal relationship in a system) and the engineering model (optimizing the system output). Moreover, even older students encounter major obstacles when they used computational modeling and simulation (Magana and de Jong, 2018).

Beyond the challenges in model-building, students have difficulties in performing system-level analyses (Basu et al., 2015, 2016b; Cheng et al., 2010; Chi, 2005; Chi et al., 2012; Wilensky and Resnick, 1999). Some of the difficulties can be attributed to the confusion in the relationship between components at different levels of abstraction. For example, emergent behaviors arise as higher-level patterns derived from aggregated interactions of lower-level components (Jacobson and Wilensky, 2006). In most cases, the relation between low-level interactions and emergent behaviors are not easily apparent because the aggregating relations are nonlinear. As a result, students' difficulty in comprehending emergent behaviors is attributed to the slippage between levels (Wilensky and Resnick, 1999).

Finally, other forms of challenges can be attributed to students' commonsense conceptions of systems used to construct their understanding of the content. Novice learners are prone to assigning false intentionality to individuals in a system (Chi et al., 2012), or assuming the existence of centralized control (Cheng et al., 2010). Because students' conceptions are often formed by induction or abduction based on causal reasoning, misconceptions are difficult to mitigate and tend to remain robust (Chi, 2005). Overall, the challenges that novice learners face are manifold, consisting of content-related difficulties, representational and system-level challenges, and the lack of efficient and effective learning strategies to overcome these difficulties.

2.4.3 Open-ended Learning Environments (OELEs)

OELEs are a class of computer-based learning environments (CBLEs) that adopt a constructivist epistemology to support the acquisition of knowledge and skills (Land, 2000). OELEs typically provide learners with opportunities to practice problem-solving skills in real-world contexts (Wang and Hannafin, 2005). A learning environment can be deemed as *open-ended* if the student working in that environment has the freedom to choose "the learning goal, the means to support learning, or both" (Hannafin et al., 2014, p. 641). Furthermore, OELEs may provide tools and resources that engage students in activities, such as generating hypotheses, constructing solutions, verifying the hypotheses with tests, and revising hypotheses in different phases of learning (Land, 2000). These processes are indispensable components of the NGSS Framework that support integrated STEM learning. For example, the MoDeLS project (Schwarz et al., 2009) provides a learning progression of more sophisticated levels of engagement and the knowledge of scientific modeling practices. The core scientific practices were made more accessible and meaningful for upper-elementary and middle school students.

Historically, intelligent learning environments (including the conceptual predecessors of modern OELEs in the 1980s) had been criticized for their inability to support students' deep learning (e.g., Ohlsson (1986)). The weakness of OELEs with respect to the limited range of teaching strategies compared to human expert teachers has been improved to a great extent by the enrichment of learning activities, deeper understanding of human teaching practices, providing more choice to students in their learning and problem-solving activities, and by incorporating advances in modeling and scaffolding technologies (Du Boulay and Luckin, 2001). In addition, OELEs have also become a popular medium to conduct strategy-related research (Panadero et al., 2016). Examples of OELEs that incorporate trace analysis for understanding students' learning strategies include Ecolab (Luckin and du Boulay, 2016), nStudy (Winne and Hadwin, 2013), MetaTuto (Azevedo et al., 2010), and Betty's Brain (Biswas et al., 2005, 2016).

VanLehn (2013) outlined *modeling* activities as (1) constructing a model instead of exploring with a pre-built model; (2) expressing the representation in a formal language instead of drawings or texts; and (3) making predictions by executing the model on a computer. Therefore, activities such as making simulations[1], drawing pictorial models, or creating concept maps were not considered *learning-by-modeling* activities because they did not involve the construction of a model (exploration-only), does not use a formal language (pictorial representations), or the artifacts created by the students were not executable (concept maps).

In addition, learning by modeling activities generally involve

- *making systematic observations and/or collecting experiences about the phenomenon under study*;
- *constructing a model of the phenomenon based on these observations and experiences*;
- *evaluating the model against standards of usefulness, predictive power or explanatory adequacy*, and
- *revising the model and applying it in new situations* (Louca and Zacharia, 2015, p. 193, italics added)

Examples of OELEs that adopt the *learning-by-modeling* paradigm include Ecolab (Luckin and du Boulay, 2016), Betty's Brain (Biswas et al., 2005, 2016), CT-STEM (Jona et al., 2014; Swanson et al., 2018), ViMAP (Sengupta et al., 2015), CTSiM (Sengupta et al., 2013; Basu et al., 2017), and C2STEM (Hutchins et al., 2020, 2019a, 2018). These learning environments can be categorized as learning-by-modeling OELEs because they allow

[1]Making simulations is an important aspect in model-building but itself does not suffice to be a modeling activity.

learning activities summarized by (Louca and Zacharia, 2015) and enable the learners to choose their own learning goals and/or the method to achieve their goals (Hannafin et al., 1999, 2014), aligned with the VanLehn definition. In the rest of this subsection, we review a subset of these the OELEs with an emphasis on how they support students' different aspects of learning and measure learning performance.

2.4.4 Ecolab

The Ecolab family of constructivist learning environments focuses on middle school's science topics, such as food chains and food webs (Luckin and du Boulay, 2016). Learners using Ecolab can model different organisms on the food chain, and explore the relationships between these organisms without needing to deal with the complexity of the entire food web (Luckin and du Boulay, 2016). Ecolab's modeling tasks utilizing the learner's Zone of Proximal Development (ZPD, (Vygotsky, 1978)) to facilitate the learning. A variation of the environment, M-Ecolab, adapts to the learner's motivational state to determine the form of scaffolding provided to the student (Mendez et al., 2005)). The environment also supports the development of metacognitive, motivational, and goal-orientation strategy constructs, in particular, providing scaffolds on the metacognitive strategies of help-seeking, task selection, and self-monitoring (Luckin and du Boulay, 2016; Mendez et al., 2005; Luckin and Hammerton, 2002). Studies with Ecolab II have shown that it supported students' learning of domain content and helps improve their metacognitive processes of task selection and self-monitoring. This is especially true for low-ability students who may lack prior knowledge or metacognitive skills (Luckin and Hammerton, 2002; Harris et al., 2009).

2.4.5 Betty's Brain

The Betty's Brain learning environment (Biswas et al., 2005, 2016; Leelawong and Biswas, 2008) is an OELE that helps students acquire knowledge and understanding of scientific phenomena, such as climate change and human body thermoregulation, by constructing causal models. Students construct the causal map to teach a virtual teachable agent, generically named Betty[2]. As she is being taught a particular topic, for example, the causes and effects of human activities that result in the greenhouse effect, and the changes in climate caused by the greenhouse effect, Betty can answer queries such as *If deforestation increases, what will happen to the amount of heat trapped by the earth?*. To answer the question, Betty uses the current causal map she has been taught to follow a succession of causal links and derive her answer to the question. Questions can be posed to Betty individually, or in the form of a quiz that is administered by the mentor agent, Mr. Davis. The mentor agent grades the quiz, and Betty's performance on the quiz provides students with contextualized feedback that they can use to check and correct their maps. Methods for building, checking, and correcting the map requires a number of strategies that students can employ to find and correct errors in the model. Studies demonstrate that students learning is directly proportional to how well they can teach Betty the correct causal map (Biswas et al., 2016; Kinnebrew et al., 2017). The mentor agent, Mr. Davis observes students' model-building and model checking behaviors, and intervenes with help on appropriate strategies when they are not performing well. A series of middle school classroom studies with the Betty's Brain system have demonstrated that students achieved significant pre- to post-test learning gains on science content (Leelawong and Biswas, 2008; Segedy et al., 2015b; Biswas et al., 2016), and students with higher learning gains and model scores (i.e., more correct links in their model) use more effective learning strategies (Kinnebrew et al., 2014; Kinnebrew et al., 2017; Munshi et al., 2018). As an OELE, Betty's Brain provides students with rich opportunities to practice solving complex and open-ended problems (Segedy et al., 2015b).

[2]The system technically adopts a *learning-by-teaching* paradigm (Biswas et al., 2005; Leelawong and Biswas, 2008) instead of *learning-by-modeling* per se. On the other hand, students' modeling activities are under the disguise of their teaching practice with the teachable agent. Therefore, this dissertation categorizes Betty's Brain as a learning-by-modeling environment as well. In addition, the causal map in Betty's Brain is fundamentally different from a concept map as discussed in VanLehn (2013) because the causal map can be *executed* to make logical inferences.

2.4.6 CTSiM

The CTSiM OELE adopts a systems-thinking approach supported by important CT concepts and practices to help middle school students learn and reason about scientific phenomena (Basu et al., 2014, 2013, 2017; Zhang et al., 2017). CTSiM employs a learner modeling and adaptive scaffolding framework that supports synergistic learning of science and CT for middle school students (Basu et al., 2017). Much like other OELEs, CTSiM offers learners the freedom to decide their own learning goals and learning trajectories that demonstrate their choice of information acquisition, model building, and model checking tasks (Basu et al., 2017). In CTSiM, students can perform five primary learning-oriented tasks (Basu et al., 2017): (1) read the built-in hypertext resource library to acquire information of domain content, and programming and CT-related concepts; (2) build conceptual models of the system by defining the components (e.g., agents and the environment) using an agent-based modeling framework; (3) construct block-based executable computational models that define the agents' interactions with the environment and with each other; (4) run the computational models as NetLogo (Wilensky, 1999) simulations to analyze and debug the agents' behaviors; and (5) compare the behaviors of their computational models to the behaviors generated by an expert reference model (the students never get to see the code for the expert model). While performing modeling activities in CTSiM, students gain the understanding of the domain content as well as CT concepts, such as variables, loop structures, and program conditionals and CT practices, such as systems thinking, problem decomposition, debugging and testing (Basu et al., 2016b; Weintrop et al., 2016a; Zhang et al., 2017).

2.4.7 C2STEM

Hutchins et al. (2019a, p. 116, italics added) pointed out that "*introducing computational modeling into STEM classrooms can provide opportunities for the simultaneous learning of computational thinking (CT) and STEM*." The Collaborative, Computational

STEM (C2STEM) learning environment builds on the conceptual framework of supporting the synergistic learning between STEM and CT for advancing STEM domain learning and developing the understanding of important CT concepts and practices (Hutchins et al., 2020, 2019a, 2018). C2STEM adopts a novel paradigm that combines visual model building and a domain-specific modeling language (DSML) to scaffold physics learning using a learner-centered and classroom-centered design approach (Hutchins et al., 2018, 2020). In C2STEM, students learn by building models that control the motion of objects. Their learning is further supported by scaffolded tasks and embedded formative assessments that introduce them to physics and CT concepts (Hutchins et al., 2020). Numerous semester-long classroom studies involving hundreds of high-school participants using the C2STEM system have demonstrated significant improvement in learning. Moreover, the C2STEM research team has developed methods to characterize students learning behaviors and correlate them with students' learning gains in the STEM and CT domains (Hutchins et al., 2020). C2STEM also includes preparation for future learning assessments that have demonstrated students' abilities to generalize and apply CT and science concepts and practices across problem-solving tasks and domains (Hutchins et al., 2020). Recently, C2STEM has been extended to include marine biology curriculum units for middle school students (e.g., (Hutchins et al., 2019b)).

2.5 Learning Strategies

When constructing models, as summarized by VanLehn (2013), students also face challenges with poor problem-solving strategies. The term *strategy* comes from ancient Greek that means plans for winning a war. In the context of educational research, strategies generally refer to the systematic plan segments for achieving goals (Oxford, 2011). In its simplest form, strategies are collections of *conditional* constructs that can be represented as computational *if-then-else* rules (Winne et al., 2002).

Better-performing students are known to be more strategic in their approach to learning than less competent learners (Derry, 1990). There is also evidence that explicit instruction of cognitive and metacognitive strategies improves the overall learning performance (Weinstein and Meyer, 1994; Cornford, 2002; Zhang et al., 2014).

By and large, strategies can be linked to metacognitive processes (Flavell, 1979) designed to foster success in the context of learning (Weinstein et al., 2011; Panadero and Alonso Tapia, 2014); some definitions also extend to the use of strategies from regulating cognitive processes to include motivation and affect. The cognitive and learning sciences literature describes strategies as a crucial component of critical thinking and learning (Boekaerts, 1996; Zimmerman, 2000). Therefore, strategies are inextricably linked to metacognition. Furthermore, the use of learning strategies helps learners become more self-reliant and fosters life-long learning (Cornford, 2002; Weinstein and Meyer, 1994).

Strategies capture procedural knowledge for accomplishing tasks and goals and are typically represented by *sequences* of activities as opposed to *single* events or actions (Mayer, 1988; Pressley et al., 1989; Garner, 1988; Alexander et al., 1998). They are known to positively influence information processing and development of new skills in support of learning (Mayer, 1988; Pressley et al., 1989; Weinstein et al., 1988; Alexander et al., 1998; Oxford, 2011). Just as learning strategies influence how students process information and learn, the effective use of learning strategies also requires proper and prompt control and regulation of the learning process (Garner, 1988; Alexander et al., 1998).

More specifically, to become effective strategy users, learners need to have adequate descriptive, procedural, and conditional knowledge of the strategies they apply (Anderson, 1983; Weinstein et al., 2011; Winne and Hadwin, 1998). This implies that learners need to acquire the processes associated with strategy execution along with methodologies for organizing, retrieving, and applying these processes. As stated by Garner (1988, p. 64, italics added), "*knowing when to use a strategy is as important as knowing how to use it.*"

Two closely related constructs, namely *cognitive* and *metacognitive* strategies, form an important part of the learning process, and both constructs are associated with orchestrating cognitive resources and skills. These two constructs differ in their generality and purpose (Weinstein and Meyer, 1994; Cornford, 2002). Typically, cognitive strategies are goal-directed, intentionally invoked, situation-specific, and not universally applicable (Weinstein and Meyer, 1994). On the other hand, metacognitive strategies involve processes such as planning, monitoring, and evaluating, which are more generally applicable (Donker et al., 2014; Cornford, 2002).

Operationally, cognitive strategies know objects and operate on objects (Winne, 1995). The term *objects* in Winne's framework loosely refers to *information*, which can be interpreted as knowledge and skills. *Metacognition* is often conveniently defined as *thinking about one's own thinking* (Flavell, 1979). However, this generic description of metacognition is "*variable and ambiguous*," especially in the context of learning strategies (Nisbet and Shucksmith, 2017, p. 24, italics added). From another perspective, metacognition can be described as deliberating on the use of particular cognitive processes and how to combine them to accomplish larger tasks (Winne, 1995).

Between the two levels of cognition and metacognition, *metacognitive monitoring* serves as the bridge connecting the two levels, as it describes the processes of observing and evaluating one's own execution of cognitive processes to exercise control and improve cognition (Kinnebrew et al., 2017). As we discuss in more detail later, such complex monitoring processes involve learners' explicit use of strategies. In addition, *management* strategies, such as controlling for time and making adjustments to the environment to manage distractions, are also defined and discussed in the literature. In this paper, we only focus on the cognitive and metacognitive strategies.

There are competing views in the literature on a learner's level of consciousness, i.e., automaticity in strategy application. Automation refers to situations when "*a strategy can be executed without close metacognitive monitoring*" (Zimmerman and Moylan, 2009, p.

317, italics added). Some studies contend that with repeated practice, learners may apply strategies habitually without much conscious thought; in other words, there is not much deliberation when selecting and applying strategies during learning (Baron, 1985; Cornford, 2002). In contrast, other studies contend that a learner's behaviors are strategic *only if* the strategies are consciously selected from a possible set (Garner, 1988). In addition, strategies encapsulate conditional knowledge, which represents processes that aid in selecting, combining, or redesigning the cognitive skills that combine to make up strategies (Kirby, 1988). Because selection, combination, and redesigning are all deliberate processes, a strategy must remain a conscious process to the learner. Garner contended that "*though some subroutines may be learned to a point of automaticity, strategies are generally deliberate, planned, consciously engaged-in activities*" (1988, p. 64, italics added).

Furthermore, there are learning theories that suggest unifying the competing views on learner's consciousness during strategy selection and execution. For example, the learning process in Fitts's skill learning theory consists of three stages, namely, the *cognitive*, *practice-fixation*, and *autonomous* phases (Fitts, 1962). In each phase, the learner's efficiency and automaticity increase with practice, which allows the learner to seamlessly invoke deliberate regulatory tasks, such as planning, monitoring, and evaluation. Therefore, the level of automaticity in strategy use can be viewed as a spectrum rather than discrete states.

2.5.1 Strategy frameworks

The study of strategies is often contextualized within the broader framework of *self-regulated learning* (SRL) (Schunk and Greene, 2018). Strategies are an important component of SRL frameworks. For example, Boekaerts' structural model of SRL includes cognitive strategies, cognitive self-regulatory strategies, motivation strategies, and motivational self-regulatory strategies among its six major components of self-regulation (1996).

This model focuses on students' purposeful use of motivational strategies (especially emotional strategies) to achieve their learning goals (Smit et al., 2017).

Weinstein et al.'s Model of Strategic Learning states that a learner needs to have the *skill*, *will*, and *self-regulation* to effectively and efficiently implement learning strategies (Weinstein et al., 2011). In this model, skill refers to the descriptive and procedural knowledge associated with learning strategies (e.g., rehearsal, elaboration, and organization). Will is linked to the motivation and affective aspects of learning (e.g., setting short-term and long-term goals). Self-regulation helps learners manage their strategic learning at different levels (Weinstein et al., 2011). Notably, this model stresses the management role of self-regulation instead of using self-regulation as an overarching component of learning.

The Cyclical Phases Model of Zimmerman emphasizes *strategic planning* as a component of task analysis in a learner's *forethought* phase. In this phase, the learner analyzes the task, assesses the expected outcome, and then evaluates the value of the task, before activating appropriate learning strategies (Zimmerman, 2000). The model defines *task strategies* as a mechanism for self-control, with the learner dividing a task into "*its essential parts and reorganizing the parts meaningfully*" (2000, p. 19, italics added). During this phase, the learner needs to apply appropriate learning strategies (e.g., self-recording, time-management, and help-seeking) to maintain a high level of motivation and track progress towards the learning goal (Zimmerman, 2000; Panadero and Alonso Tapia, 2014).

The COPES model (conditions, operations, products, evaluations, and standards) of Winne and Hadwin describes self-regulated students as those who actively manage their learning via enacting and monitoring their own cognitive and metacognitive strategies as their learning progresses (Winne, 1995; Winne and Hadwin, 1998). The COPES model demonstrates that learning is enabled by self-regulation across four linked and looping phases of *task definition, goal setting and planning, enactment of tactics*, and *adaptations to metacognition* (Winne and Hadwin, 1998; Winne et al., 2002). In the COPES model, a strategy is defined as a collection of *if-then* rules (also known as tactics) that create larger patterns to form *if-then-else* rules over time (Winne et al., 2002). Strategies are more complex than tactics in their structure, and they also have a larger scope that yields more information that can be used as feedback during learning (Winne et al., 2002). In the recursive loop defined in the COPES model, a learner first activates the memory of previous strategy use in the task definition phase, and then the strategy is linked to specific learning goals in the goal setting and planning phase. Following that, in the enactment phase, the learner uses linked strategies to address the learning goals. Finally, in the adaptations to metacognition phase, the learner evaluates the effect of the strategy used, and then tunes or restructures the strategy to make it more effective for the goals and plans of the learning task (Winne et al., 2002).

2.5.2 Summary

As reviewed in this section, strategies that support effective learning and problem-solving have been receiving attention in the teaching-and-learning literature since the mid-1980s (Alexander et al., 1998; Nisbet and Shucksmith, 2017). There is also more recent research (e.g., Gasevic et al. (2017)) that focuses on detecting and analyzing learning strategies in CBLEs. However, the existing work on learning strategies often ignores three issues, especially for applications in CBLEs. Mainly,

1. most studies define learning strategies in a generic way as *skills or methods to enhance performance* without grounding them in operational forms. As a result, learning strategies cannot be measured on a consistent numerical scale;

2. the study of learning strategies primarily focuses on learning domains, such as reading, writing, science, and mathematics. Previous work, though invaluable in their specific domains, has not been extended to studying strategies for learning in OELEs in a generalizable manner; and

3. there is a dearth of systematic approaches for measuring and evaluating strategy use from students' activity data in OELEs.

More specifically, past studies on strategy analysis have relied on *self-report, think-aloud* protocols and *just-in-time* interviews to measure students' use of learning strategies (Winne et al., 2002; Schunk and Greene, 2018; Vermunt, 2020; Gasevic et al., 2017, Chapters 21, 22). These methods suffer from issues such as (1) the set of self-report items often do not form an internally consistent category to measure the latent variable (Winne et al., 2002), and (2) the interviewer interprets and represents the learner's response through their own heuristics lenses (Winne et al., 2002). In other words, the observation of a phenomenon (strategy definition and use) is sensitive to and can be greatly impacted by the data collection method (Dent, 1999). Furthermore, the lack of a systematic measurement framework also impacts the reliability of the measurement (Winne et al., 2002).

2.6 Critical Summary and Motivation for Research

As reviewed in Section 2.2 of this chapter, the projects contributing to the integration of science learning and engineering education primarily focus on higher grades in K-12 (e.g. (Kolodner et al., 2003b; Irgens et al., 2020)) and through the construction of physical, tangible artifacts (e.g., using LEGO bricks (Wendell and Rogers, 2013) or designing the human elbows (Penner et al., 1998)). As a result, the affordances of CT (Wing, 2011; Shute et al., 2017) and computer-based learning environments (de Jong et al., 2013; Land, 2000) are not fully utilized. In addition, many CT-centric projects (e.g., Scratch (Brennan and Resnick, 2012) and AgentSheets (Repenning et al., 2000)) focus on multi-media narrative or game design, any may not directly support the learning of STEM contents (Weintrop et al., 2016a; Zhang and Biswas, 2019).

On the other hand, previous work in the literature has demonstrated the potential and benefits of supporting the *synergistic* learning of STEM and CT using open-ended learning environments (e.g., (Basu et al., 2017; Hutchins et al., 2020; Mathews et al., 1989; Weintrop et al., 2016b). In the present work, synergistic learning is defined as a mutually-benefiting relationship between different learning domains that servers as the basis for the integrated learning of science, engineering, and CT.

Chapter 3

Statement of the Problem

This dissertation aims to address the dearth of frameworks and supporting empirical research that uses computational thinking as a platform to support the integrated learning of science and engineering concepts and practices. More specifically, our methods adopt a learning-by-modeling approach, where students apply CT concepts and practices to construct computational models of science phenomena, and then use the computational models to solve engineering design problems. The modeling and design problem-solving tasks are complex, and students have to invoke several cognitive processes and a combination of cognitive and metacognitive strategies to complete their tasks. Figure 3.1 illustrates this framework.

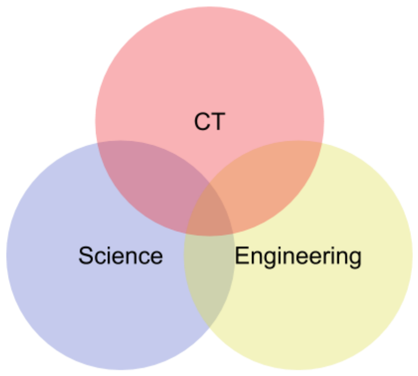


Figure 3.1: Venn diagram of the conceptual framework

As summarized in Section 2.6, previous work on the integration of science and engineering learning (e.g., Fortus et al. (2005); Brophy et al. (2008)) tend to involve upper-middle and high school students and favors the construction of *tangible* objects. As a result, the affordances of virtual learning environments and CT (reviewed in Section 2.3.2) are not fully exploited. On the other hand, many existing CT and learning-by-modeling applications focus on narrow content domains, that include multi-media narratives and games. Most of these systems do not promote the learning of domain contents and CT concepts and practices in an integrated manner (Weintrop et al., 2016a; Zhang and Biswas, 2019). As a result, there is a strong need for designing learning environments and curricula to advance the integration of engineering education with science learning, especially through CT-based platforms. This is the primary focus of the work presented in this dissertation.

In addition, the literature review on learning strategies discussed in Section 2.5, revealed two issues: (1) the lack of grounding of strategies in operational forms of students' activities in learning environments making it hard to identify and measure specific strategy use; and (2) inability to systematically measure strategy use from students' action sequences in OELEs. To address these issues, this dissertation also proposes a framework for modeling, measuring, and assessing learners' strategies in OELEs. We study students' strategy use in learning by modeling environments for science domains. In summary, the intellectual and practical problems this dissertation proposes to solve are:

1. Develops a framework for integrated science + engineering learning using CT as the supporting platform;
2. Design and implement the SPICE system that combines computational modeling with design problem-solving.
3. Study how the integrated science +engineering environment supports learning of science and engineering concepts and practices.
4. Tracking students' model-building and design problem-solving activities helps us analyze students' learning strategies, and how these strategies corresponding to students' learning and problem-solving tasks.

3.1 Research Approach and Hypothesis

This dissertation research supports the Water Runoff Challenge (Chiu et al., 2019; McElhaney et al., 2020) as part of the Scientific Projects Integrating Computing and Engineering (SPICE) research project. The WRC curriculum is a three- to four-week-long unit that is aligned with NGSS performance expectations in Earth science and engineering design. The science domain content involves Earth and Environmental Science for upper-elementary and lower-middle school curricula (grades 5-6), emphasizing the movement of surface water in a system after heavy rainfall, and the human impact of this runoff on the environment. The WRC is situated in an authentic problem-solving context where school grounds often get flooded after heavy rainfall, resulting in the playground being unusable for a period of time. Furthermore, the resulting runoff into the surrounding neighborhoods and streams can have environmentally damaging effects (provide a reference).

Students need to address the WRC by redesigning the schoolyard with different surface materials to minimize the amount of water runoff after a storm while adhering to a series of design constraints in the overall cost, accessibility, and different utilities of the schoolyard (McElhaney et al., 2020). This learning context is authentic and relevant to students facing similar problems (limited usability and pollution) in their own schools, therefore, the WRC is potentially engaging and personally meaningful to the learners.

In the WRC, students begin with engaging in a series of hands-on activities involving experimenting with physical materials commonly available in schoolyards and playgrounds[1] and then contrasting the absorption capabilities of these different surface materials. After acquiring a basic understanding of the runoff scenario, students develop conceptual, pictorial representations that express the amount of water runoff as the difference (if any) between the total rainfall and water absorbed by surface materials.

Following that, students are introduced to the CT concepts of variables, expressions, and conditionals in a series of unplugged activities as a primer of the computational modeling activity. Then, using the CT constructs provided in the learning environment, students create an executable computational model that determines the output variables in the runoff system. During the computational modeling activity, students also practice important CT skills such as testing and debugging their code to refine their models. With the correct im-

[1]The man-made materials include concrete, permeable concrete, artificial turf, and poured rubber; and the natural materials include grass, wood chips, and bare dirt.

plementation of a computational runoff model, students can simulate the effect of surface materials on runoff from the schoolyard to the surrounding area. The same computational model will then be used when students perform engineering designs by creating and testing schoolyard designs that need to meet specified constraints (Zhang et al., 2019; McElhaney et al., 2020).

The WRC is designed to support the integration of the learning of CT, engineering, and science. It also provides rich opportunities for students to apply learning strategies to facilitate integrated learning and problem-solving tasks. This dissertation hypothesizes that (1) science learning and engineering performance share a synergistic relationship, i.e., the mastery of one construct facilitates the other; (2) computational thinking has a supporting role in the learning of both constructs; (3) understanding students' use of learning strategies in these processes will offer insights and help the students become more successful learners. More specifically, the research questions are:

- *RQ 1:* How do we design and implement a computational modeling environment to support the integrated learning of science and engineering and how effective is this curriculum in supporting student learning?
- *RQ 2:* What are the relationships between science learning and engineering performance? Alternatively, how does the learning of science concepts by building computational models of scientific processes aid the engineering design process?
- *RQ 3:* What is the role of computational thinking in facilitating science learning and engineering design?
- *RQ 4:* How do students utilize strategies to facilitate their learning processes (i.e., the construction of computational models and generating engineering designs)?

To answer RQ 1, this dissertation proposes to provide a principled approach to designing and implementing computational modeling and engineering design environment based on an Evidence Centered Design (ECD) (Mislevy et al., 2003) and domain-specific modeling language (DSML) (Hutchins et al., nd) approach. Chapter 4 presents the design and implementation processes of the learning environment and its functionalities to store and process learning traces. In addition, Chapter 6 presents the learning outcomes to discuss the effusiveness of the curriculum.

To answer RQ 2, this dissertation proposes to measure students' science and engineering learning performance with NGSS-aligned assessments and apply learning analytics-based (Siemens and Baker, 2012) approaches to measure and analyze the computational building and engineering design behaviors and performances.

To answer RQ 3, the measured behaviors and performances will be modeled using Path Diagram Analysis (Wright, 1983; Pearl and Mackenzie, 2018) to investigate the facilitating and synergistic roles of CT, engineering design, and science learning. Results from the extensive analyses are presented and discussed in Chapter 6 to answer RQs 2 and 3.

To answer QR 4, this dissertation proposes to track students' development, application, and evolution of learning strategies as they work in the WRC using Markov Chain modeling (Russell and Norvig, 2003). The students' strategy use is presented and discussed in Chapter 7.

3.2 Operationalization

To answer the research questions proposed in this dissertation, we first operationalize the key constructs in the conceptual framework (visualized as Figure 3.1) and provide a brief introduction to the measurements of these constructs.

- Science learning: students' performance in (1) the NGSS-aligned assessment of science understanding performance expectations (PEs) used as pre-post assessments and (2) the formative assessments administered as homework during the curriculum implementation.

- Engineering design: students' performance (1) the NGSS-aligned assessment of engineering PEs used as pre-post assessments, (2) students' performance in the schoolyard design challenge measured by the quality of each engineering design, and empirical evaluation of the systematicity of the design process.

- Computational Thinking: a collection of concepts and practices (not considering the perspectives) involved in problem-solving using computational models measured by (1) summative assessments and (2) mastery and usage of CT while creating computational runoff models.

- Learning Strategy: students' conscious and controllable action sequences to facilitate and enhance task performance measured by theory-driven and data-driven analytical methods.

Chapter 5 discusses the methods used in the research in greater detail and provides justifications on how these constructs are defined and measured in a teacher-led classroom study with sixth-grade students. In addition, we revisit the operationalized constructs in the context of data collected from the classroom study in Chapter 6.

Chapter 4

SPICE: Computational Modeling and Engineering Design Environment

The SPICE WRC research project adopts *agent-based modeling* concepts (Collins and Ferguson, 1993) to support the decomposition of the dynamic processes into its meaningful constituent parts. Furthermore, to be compatible with middle school math proficiency, typical differential equation representations for systems dynamics are simplified to discrete-time algebraic forms (Sengupta et al., 2013; VanLehn, 2013).

4.1 The Runoff System and Its Representations

The *amount of runoff* after a rainfall is a key construct in the WRC. Traditionally, the total runoff rate of a small watershed is calculated by the *Rational Equation*, a widely-applied method in hydraulic engineering to estimate the peak discharge (Thompson, 2006). The equation for peak discharge volume is $Q = c \times i \times A$, where c is a unit-less runoff ratio, i is the rain intensity, and A is the drainage area.

With the Rational Equation, the runoff scenario can be modeled by *system-dynamics* representations that predict the behavior of a system over time (VanLehn, 2013). The amount of absorption, runoff, and the total amount of rainfall is computed as:

$$total_{rainfall} = \int i \, \mathrm{d}t \tag{4.1}$$

$$total_{runoff} = \int ci \, \mathrm{d}t \tag{4.2}$$

Figure 4.1 shows a pictorial representation of variables and their relationships in the runoff system in the form of a computation graph (computation graph is defined in (Browne, 1986)). Notation wise, (1) the yellow rectangles represent user-defined input to the system; (2) the green rectangles represent intermediate variables of the runoff model; (3) the blue rectangles represent the integral of rates (Equations 4.1 and 4.2); (4) the clear rectangles represent the output variables of the system (total runoff, cost, and accessibility); and (5) the arrows indicate a relationship in the model. For example, the total runoff is calculated as the difference between the total rainfall and total absorption to maintain the conservation of matter in the system.

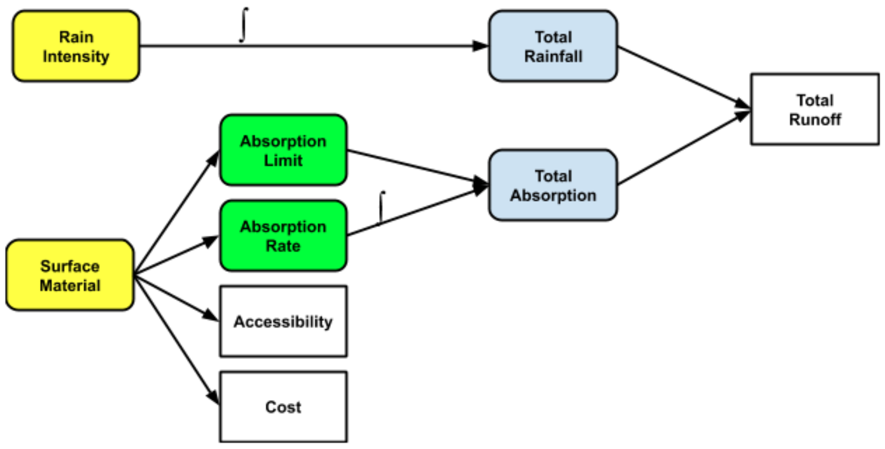


Figure 4.1: System equations representation of the runoff problem

Calculus is not part of the upper elementary or middle school curricula in the United States, so the system dynamics model for water runoff has to be simplified for these students. For example, time can be simplified to a discrete quantity as opposed to a continuous-valued variable in the differential equation (van Joolingen et al., 2005; VanLehn, 2013). In other words, we create a level of abstraction for the middle school curriculum, and the integral equations can now be represented in algebraic form:

$$total_{rainfall} = i \times \Delta t \tag{4.3}$$

$$total_{runoff} = c \times i \times \Delta t \tag{4.4}$$

Equations 4.3 and 4.4 are simpler and more accessible for younger learners and meet the middle school guidelines for math standards.

Algorithm 1 provides a pseudo-code implementation for computing the amount of total rainfall, absorption, and runoff using the simplified system dynamics model illustrated in Figure 4.1.

Algorithm 1: Time-based *system dynamics* implementation

```
Result: total runoff
rain intensity := $user input$;
duration := $user input$;
total rainfall := 0;
total absorption := 0;
total runoff := 0;
time elapsed := 0;
initialize absorption rate with current schoolyard design;
initialize absorption limit with current schoolyard design;
while time elapsed $\leq$ duration do
    total rainfall += rain intensity $\times$ $\Delta$t;
    total absorption += min(absorption limit $-$ total absorption, absorption rate $\times$
      $\Delta$t );
    total runoff := total rainfall $-$ total absorption;
    time elapsed += $\Delta$t;
end
```

For students who are not proficient in algebra and the concept of rates, the representation can be further simplified by relaxing the time component —instead of being a *system dynamic* model, it can be represented as a conditional *rule-based* model (Collins and Ferguson, 1993). Figure 4.2 shows the result of the relaxation.

Figure 4.2 has an overall structure similar to that of Figure 4.1 with the following differences: (1) instead of calculating the *amount of total rainfall* using integral or algebraic

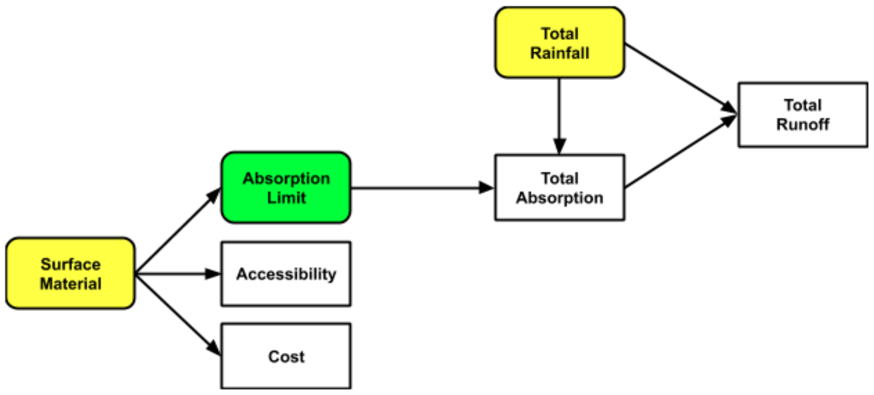


Figure 4.2: System rules representation of the runoff problem

equations, we make it an input to the rule-based model; (2) the influence of the surface material on the absorption rate is abstracted away and replaced by the total amount absorbed, which is still a function of the material being used; (3) there is an added relationship between total rainfall and total absorption, as the latter variable is determined by the smallest value of the absorption limit for a specific material and the total rainfall. This change enables the runoff system model to be represented as an aggregated model instead of a temporal model so that students with grade-appropriate math knowledge can make sense of the modeling representations and build their runoff models. The three variables (total rainfall, total absorption, and total runoff) in Figure 4.2 are tightly constrained by the conservation equation, which enforces the fact that the amount of total runoff is solely determined by the amount of rainfall and the material being used.

Algorithm 2: Simplified *rule-based* implementation

```
Result: total absorption, total runoff
total rainfall := $user input$;
initialize absorption limit with current schoolyard design;
total absorption := min(total rainfall, absorption limit);
total runoff := total rainfall − total absorption;
```

Algorithm 2 provides a pseudo-code implementation of the runoff model implemented as a compact rule-based system. To match grade-level CT concepts and algorithmic skills of lower middle school students, the computation in Algorithm 2 can be unpacked in the explicit form as in Algorithm 3. This is the science model we expect students to build to learn water runoff concepts.

Algorithm 3: Simplified, explicit *rule-based* implementation

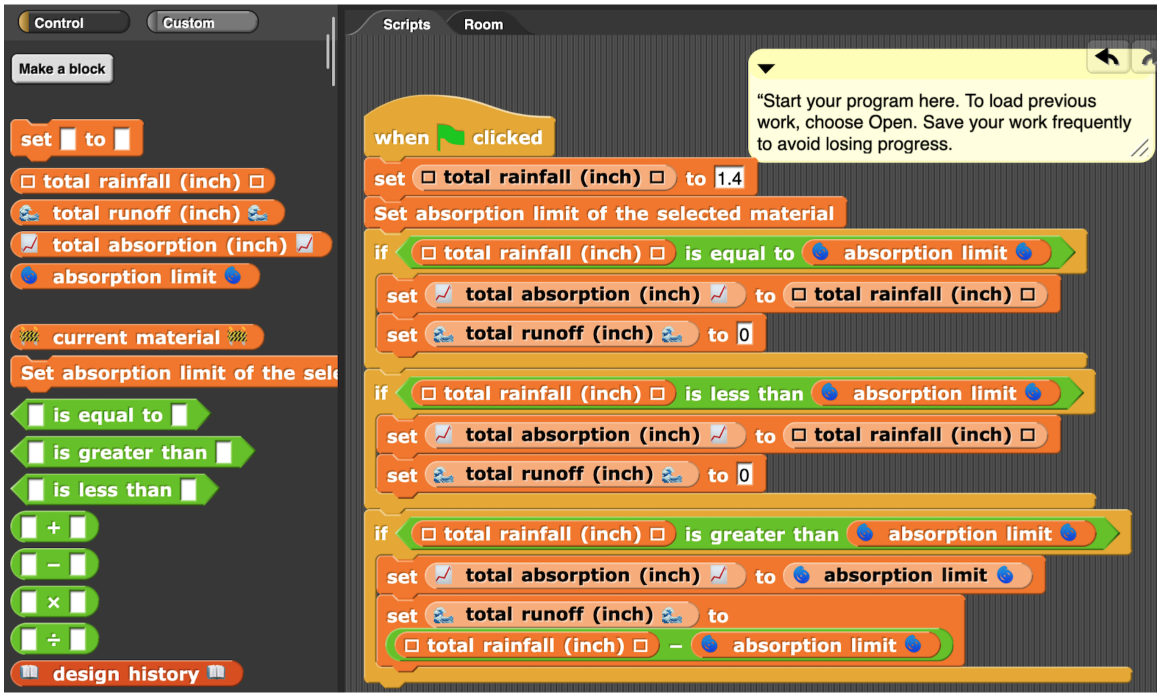
```
Result: total absorption, total runoff
total rainfall := $user input$;
initialize absorption limit with current schoolyard design;
if total rainfall == absorption limit then
    total absorption := total rainfall;
    total runoff := 0;
end
if total rainfall < absorption limit then
    total absorption := total rainfall;
    total runoff := 0;
end
if total rainfall > absorption limit then
    total absorption := absorption limit;
    total runoff := total rainfall − total absorption;
end
```

4.2 The Domain-Specific Modeling Language (DSML) and Model Implementation

Students (and classroom teachers who may not be proficient in computing and CT concepts) learn the science concepts related to the runoff system by explicitly building the model illustrated in Figure 4.2. To facilitate this, we make the modeling representation explicit and linked to the science concepts. We have created a domain-specific modeling language (DSML) to support students' computational modeling activities (Hutchins et al., nd; Zhang et al., 2019). The use of a DSML simplifies the modeling task by adapting the level of system abstraction to an appropriate level for the learners. Compared to using a general-purpose language to develop scientific models, the DSML enables students to concentrate on the science concepts and system variables without having to attend to the details of lower-level programming constructs and their syntax (van Deursen et al., 2000; Hutchins et al., nd). In the current curriculum, students first develop the computational model of water runoff, and subsequently design and test solutions to mitigate the runoff problem (McElhaney et al., 2020).

Figure 4.3 shows the DSML created for the computational modeling activity and a correct implementation of the runoff model that represents Algorithm 3 in Section 4.1. The DSML is created as a layer on top of the NetsBlox visual programming environment (Broll et al., 2018; Broll, 2018). It adopts CT concepts and practices (Wing, 2006; Sengupta et al., 2013; Shute et al., 2017) along with the primary domain concepts of (1) the amount of rainfall, (2) absorption of water by different surface materials, and (3) runoff to support the modeling of the water runoff processes. In addition, the DSML specifies key arithmetic and algebraic mathematical operations and computational constructs, such as conditional structures to support model building.

As noted earlier, system dynamics are simplified by abstracting away the concept of time and rates of change (see Algorithm 3). One should note that the WRC can be easily extended to higher grades with more complex modeling activities by changing the DSML structure to include additional concepts and mathematical constructs, e.g., time-based models as Algorithm 1.



"Start your program here. To load previous work, choose Open. Save your work frequently to avoid losing progress.

```
when [flag] clicked
set total rainfall (inch) to 1.4
Set absorption limit of the selected material
if total rainfall (inch) is equal to absorption limit
    set total absorption (inch) to total rainfall (inch)
    set total runoff (inch) to 0
if total rainfall (inch) is less than absorption limit
    set total absorption (inch) to total rainfall (inch)
    set total runoff (inch) to 0
if total rainfall (inch) is greater than absorption limit
    set total absorption (inch) to absorption limit
    set total runoff (inch) to
        total rainfall (inch) − absorption limit
```

Figure 4.3: DSML of the runoff model and a correct implementation

Using the DSML blocks, students create a rule-based computational model that defines absorption and runoff by determining the amount of rainfall that can be absorbed by the surface material and the actual amount of rainfall (see the example implementation). A visual interface allows the students to populate individual squares, and the system calculates the total runoff as well as the cost associated with designing the schoolyard with the selected materials (Zhang et al., 2019; Chiu et al., 2019). Figure 4.4 depicts the runoff simulation interface. The first image shows a single square being tested with 1.4 inches of total rainfall, causing 0.4 inches of runoff.

Table 4.1: Material information for the engineering design activity

material	absorption limit	cost	accessible
concrete	0.1 inches	$37,500	yes
permeable concrete	1.3 inches	$93,750	yes
natural grass	1.2 inches	$18,750	no
engineered wood chips	1.0 inch	$37,500	no
artificial turf	0.6 inches	$112,500	yes
poured rubber	1.2 inches	$187,500	yes

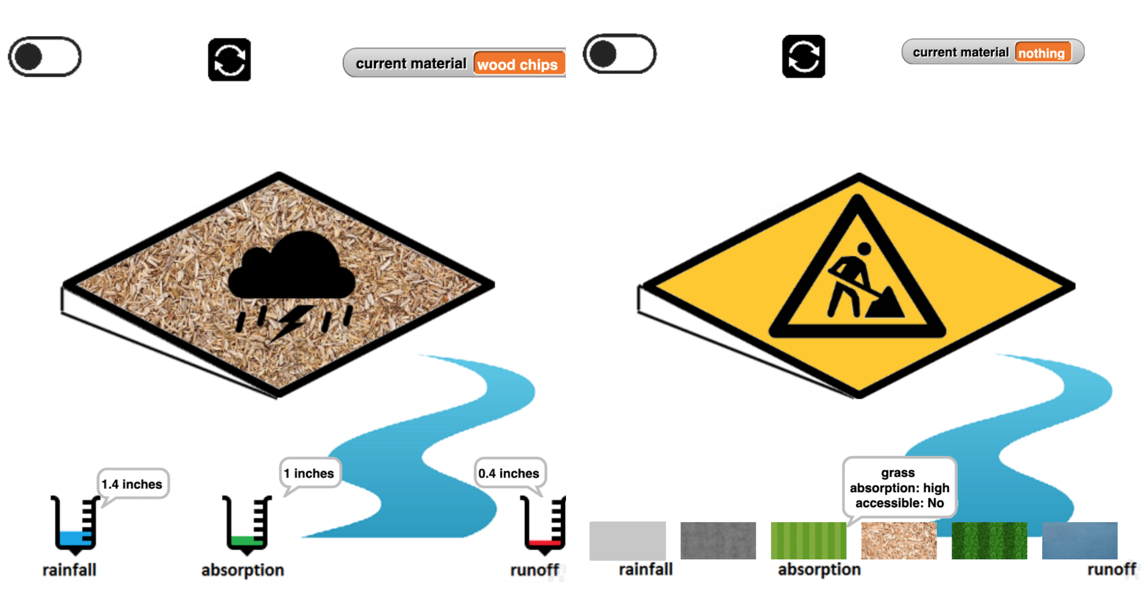


Figure 4.4: The runoff simulation and material selection interface

This design problem is challenging for middle school students. Typically, the more absorbent and accessible materials tend to have higher costs (detailed properties of all 6 materials listed in Table 4.1[1]), so students need to analyze the trade-offs between cost, absorption, and accessibility in looking for “optimal” design solutions. A non-systematic *trial-and-error* approach may overwhelm the student’s search processes as the entire problem space consists of 6^{12} possible designs[2].

[1]The costs and the values of the absorption limit are validated and provided by collaborators at SRI International and the University of Virginia.

[2]Four squares on the 4×4 grid are non-modifiable school buildings.

After constructing a correct computational model, the students use their own models in the engineering design task using an extended version of the runoff model under the hood to support the 4×4 schoolyard designs interface[3]. The students go through a search process for a solution by systematically finding appropriate materials for the schoolyard to minimize the runoff. In addition, the engineering design solution needs to satisfy the criteria of (1) having a total cost of the design $\leq \$750,000$; (2) having at least 6 squares of the schoolyard with accessible materials for those with wheelchair needs; and (3) minimizing runoff.

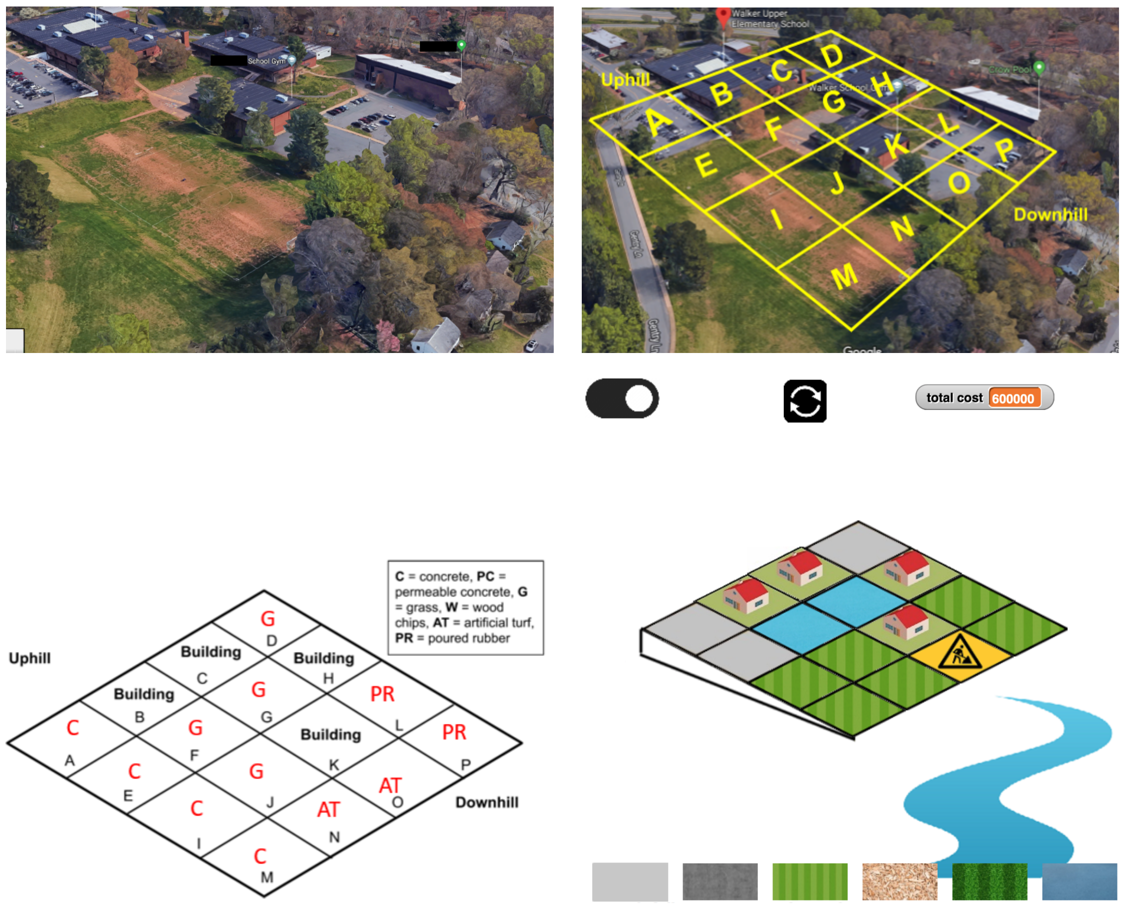

[redacted]
School Gym
[redacted]
Walker Upper Elementary School
Uphill
A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P
Downhill
total cost 600000
C = concrete, PC = permeable concrete, G = grass, W = wood chips, AT = artificial turf, PR = poured rubber
Uphill
Building
Downhill

Figure 4.5: Google Maps^{TM} views of the schoolyard (top row) and the abstract representations of the engineering design interface (bottom row)

[3]Each square has an area of 37,500 sq. ft.

Figure 4.5 shows the terrain of the school where the runoff problem occurred (top left). A four-by-four grid was placed on top of the Google Maps^{TM} view (top right). The bottom left image in Figure 4.5 shows a plain abstract representation of the existing schoolyard surface. Finally, the image on the bottom right shows a schoolyard design being built in the NetsBlox environment. The yellow (unassigned) square on the schoolyard can be filled with any of the six available materials shown on the bottom (see Table 4.1).

A National Research Council (2009) report and Moore et al. (2015) highlighted important aspects for engineering design processes of (1) having a meaningful learning context, (2) being iterative, (3) being open-ended (having multiple possible solutions of a problem), and (4) enabling systems thinking, modeling, and analysis. The computational modeling and engineering design environment designed and implemented for the WRC meets all four principles. To begin with, the learning tasks are well situated in an authentic problem that many students living in the eastern region of the U.S. may face. Secondly, both the computational model-building and schoolyard design activities are highly interactive as students need to arrive at a model or a design by systematically testing the current solution and refining it. Thirdly, the system does not reinforce a single correct solution to both learning activities, and there exist isomorphic or alternative solutions in both activities.

4.3 Interactive Support and Logging

The synchronous interactive components in the engineering design activity are prebuilt for the students in the computational models of other Sprites in the NetsBlox project (e.g., schoolyard squares and materials). The synchronous communication between Sprites was implemented as interactive events (i.e., mouse clicks). With this modularized design, learners can concentrate on implementing the core component of the runoff system.

Figure 4.6 presents the flow charts of the interactive `request-response` event handling processes of the Schoolyard Square and Material sprites. To begin with, a square sprite has flip-flop states of `constructed` and `unconstructed`. When a constructed square is clicked, the current construction status will be removed and the cost of the current material on that square will be subtracted from the total cost. Otherwise, the unconstructed square will broadcast a `SelectMaterial` event and be blocked until the event is handled by one of the *Material Sprites*. Meanwhile, the Material Sprites will stay hidden until receiving a `SelectMaterial` event from a Square Sprite. Clicking on the Material Sprite will cause the cost, runoff limit, accessibility, and other information to be recorded and sent back to the square sprite as a response. After `SelectMaterial` is handled, the square will update its construction state and then changes its look accordingly (cf. the bottom right image in Figure 4.5).

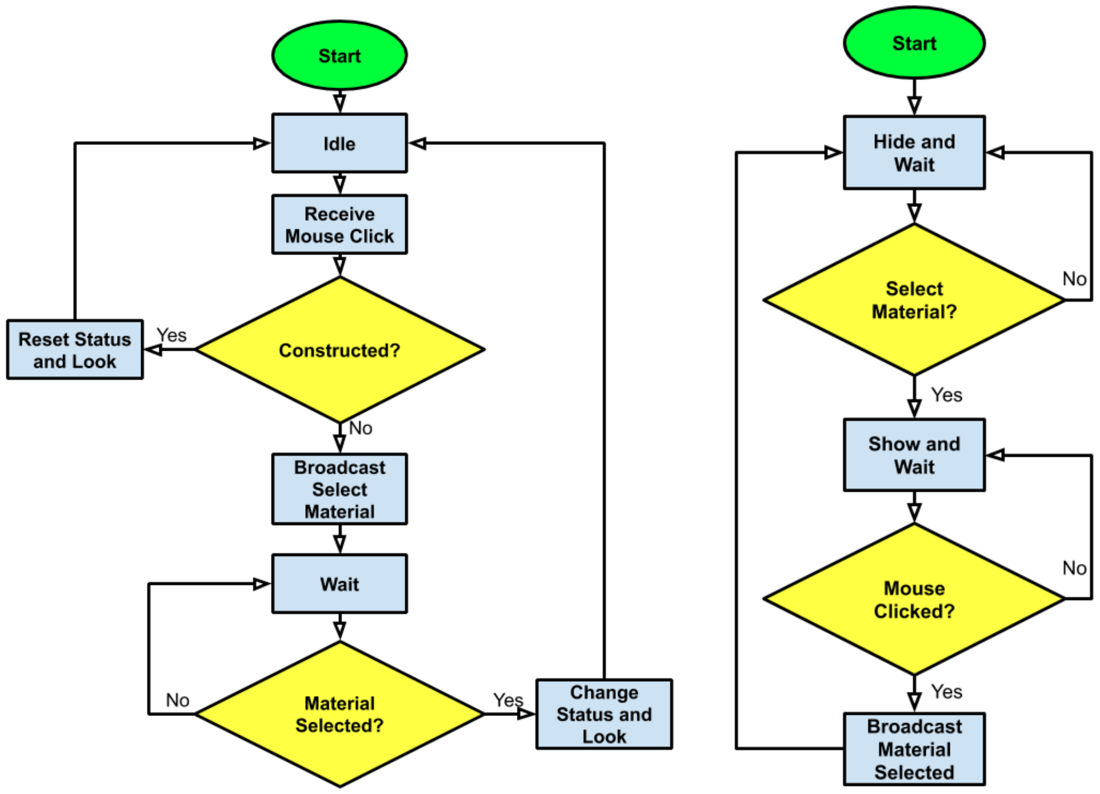


Figure 4.6: Flow charts of sample interactive event handling processes of the schoolyard square sprite (left) and the material sprite (right)

The `request-response` interaction is one example of the hidden interactions in the WRC environment[4]. Other hidden interactions include

1. A series of Remote Procedure Call (RPC) (Birrell and Nelson, 1984) to invoke server utilities such as using `Gnuplot` to visualize simulation results (e.g., the "rain plots" reported in (Zhang et al., 2019));

2. The recording and display an entire history a student's design results and simulation results using the Cloud Variables RPC (Broll, 2018) (cf. the `design history` block in Figure 4.3)

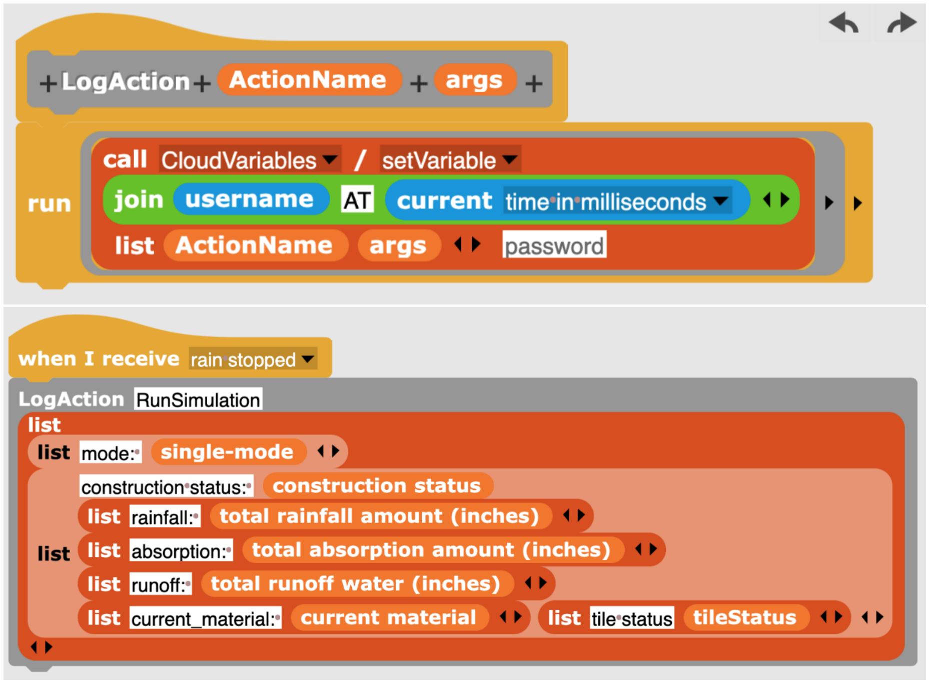
3. Logging timestamped student actions. Figure 4.7 shows the `LogAction` function implemented with the NetsBlox RPC (top) and an example action `RunSimulation` logged with the function (down). In addition, the next section presents the type and analyses of these log data in greater detail.

4.4 Methods and Techniques for Log Analysis Using Traces

The exploration of students' learning and problem-solving processes in OELEs is linked to understanding the "*dynamic and cyclical nature of self-regulation*" processes they employ (Schmeck, 2013, p. 5). Wittgenstein (1968, Sec. 530) stated that "An 'inner process' stands in need of outward criteria". As a form of *outward criteria*, the traces of students' actions, i.e., students' observable behaviors as they engage in learning tasks in an OELE, provide information about their learning processes. These methods, based on students' activity sequences, are beginning to supplant traditional measurements of strategy use using self-reports, think-aloud, and interviews (Schmeck, 2013).

The increased deployment of OELEs has made it possible to collect detailed and fine-grained traces of students' interactions and their outcomes within the learning environ-

[4]The implementations were made invisible to the students in the project to prevent meddling or confusion. During classroom studies, the hidden program were shown and explained to attentive students were also more proficient in programming on a personal basis.



```
+LogAction+ ActionName + args +
run call CloudVariables / setVariable
    join username AT current time in milliseconds
    list ActionName args   password

when I receive rain stopped
LogAction RunSimulation
list
  list mode: single-mode
  list
    construction status: construction status
    list rainfall: total rainfall amount (inches)
    list absorption: total absorption amount (inches)
    list runoff: total runoff water (inches)
    list current_material: current material   list tile status tileStatus
```

Figure 4.7: The `LogAction` function implemented with the NetsBlox cloud variables RPC (top) and an example action `RunSimulation` logged with the function (bottom)

ment (Schmeck, 2013). A series of methods have been designed to infer students' use of strategies by distilling contextualized and operationalized information from the raw traces, for example, (Azevedo et al., 2013; Bannert et al., 2014; Hadwin et al., 2007; Kinnebrew et al., 2013; Kinnebrew et al., 2017; Segedy et al., 2015b; Winne et al., 2002). These analytical frameworks or methods lay the foundation for tracing and evaluating how students' use of strategies evolve through the intervention.

The NetsBlox environment (Broll, 2018) collects the lowest-level of user action data as clickstreams and the changes to the computational blocks. On the other hand, the primitive logging mechanism posts three difficulties to analyze students' learning activities:

1. The action history was associated with a project file. It's possible to have missing log data when a student forgets to save the project.

2. The log data do not directly reflect students' intent or the context of the edit. More specifically, the log data reflect what happened to the blocks, instead of what/why the

student performed certain actions. For example, the log action `setBlockPosition` can refer to two different actions of a) dragging the block(s) around on the canvas or b) disconnect the block(s).

3. The project file only contains information about the latest version of the student's code/model. As a result, the trajectory of the modeling-building process is lost. This is due to the overhead of saving a project in the NetsBlox environment.

This dissertation research addresses the above-mentioned issues by adding a data analysis framework that extends the primitive NetsBlox logging mechanism. The framework provides a Service Layer (Papazoglou and Georgakopoulos, 2003) to track and analyze students' model-building and design processes. It supports (1) tracking students' model-building progressions (how did they construct their models step by step) and (2) evaluating the accuracy/completeness of their computational models. This new data analysis component of the computational modeling activity extends our previous work (Zhang et al., 2019) to track and analyze students' engineering design processes with the `Cloud Variables` RPC (cf. Section 4.6).

Figure 4.8 presents the data flow diagram to collect and analyze the trace data of the SPICE WRC curriculum. To begin with, raw action logs are queried from the NetsBlox NoSQL database. The first step is to aggregate the actions at the level of a (`username`, `project name`) tuple. The raw log data often contain `null` values in the fields[5]. To mitigate this issue, the framework uses a fuzzy matching rule to complete the missing `username` or `project name` fields with those that share the same `sessionId` that is created when a connection to the NetsBlox server is established. Secondly, the data preparation step processes the primitive logs and the `Cloud Variables` PRC logs to (1) generate user actions that contain the context of the behavior, (2) represent the students' computational models (and the change in each step of the process) as abstract syntax trees (ASTs),

[5]The `null` fields are often resulted from the asynchronous behavior of the `JavaScript` client code of NetsBlox.

and (3) calculate measurements of the students' behaviors and performances in the engineering design activity.

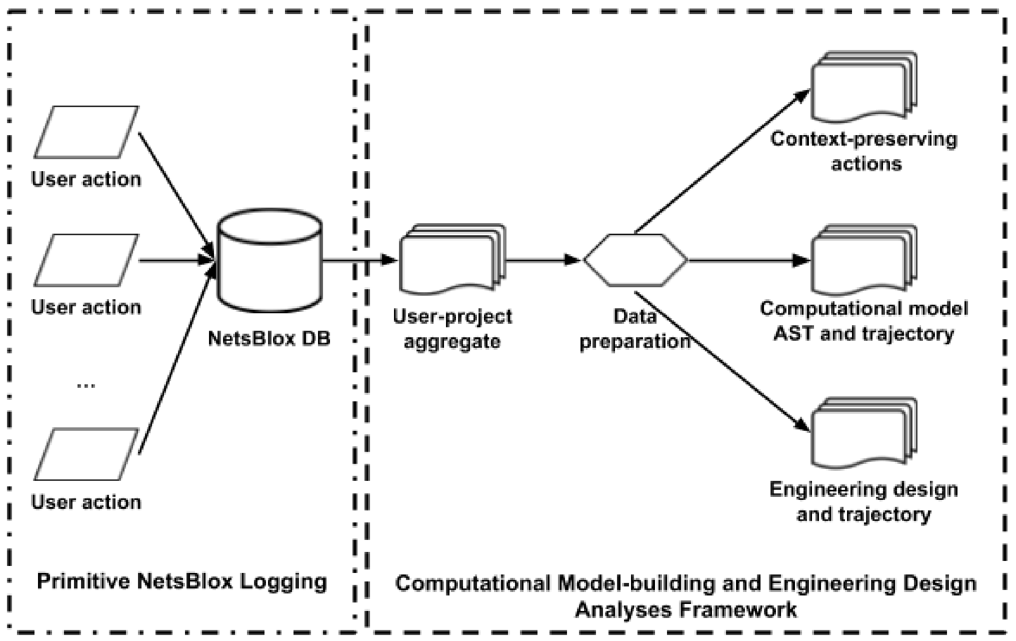

User action
User action
...
User action
NetsBlox DB
Primitive NetsBlox Logging
User-project aggregate
Data preparation
Context-preserving actions
Computational model AST and trajectory
Engineering design and trajectory
Computational Model-building and Engineering Design Analyses Framework

Figure 4.8: Data flow diagram of the data collection and analysis framework

This data collection and analysis framework resolves (1) the *loss of log data* issue by directly pulling data from the server database instead of relying on exporting the saved project file, (2) the *loss of context and student intent* issue by introducing more contexts to the primitive logs (e.g., differentiate "disconnecting a part" from "dragging things around"), and (3) the *missing learning trajectory* issue by reconstructing the processes of editing a computational model from the log data. Sections 4.5 through 4.6 presents the implementation of the computational model-building and engineering design analysis framework in greater detail.

4.5 Analyzing Computational Model-building Activities

The learning environment logs individual learners' actions during their **computational model-building** activities. It can also calculate measurements such as (1) the number of actions to add, remove, connect, or disconnect programming blocks and (2) the number of actions to test the correctness of the resulting model. More specifically, a computational block has three states (*not present*, *disconnected*, and *connected*). Depending on the state transition, the action that modifies the computational model can be inferred in the *post hoc* analyses.

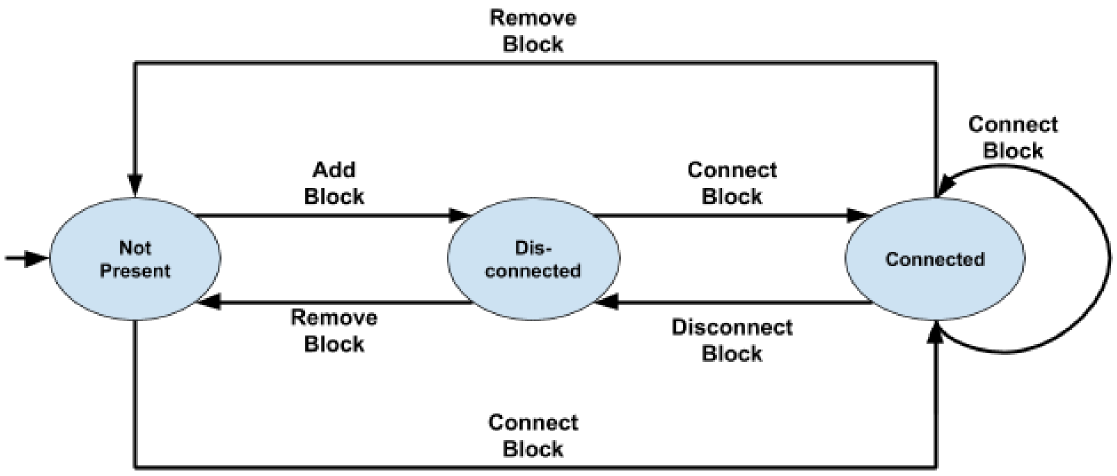


Figure 4.9: Computational block life cycle

Figure 4.9 shows the life cycle of blocks. Correspondingly, Table 4.2 shows the definition of computational model editing actions that reflect the transitions in the block life cycle diagram. These actions are `AddBlock`, `ConnectBlock`, `DisconnectBlock`, and `RemoveBlock`. Note that Table 4.2 omits a possible transition on purpose (Disconnected → Disconnected) because simply changing the position of block(s) is considered a no-operation as it does not change the semantics of the computational model[6].

We also log two other actions in the computational model-building activity: `FillField` and `StartSimulation`. As shown in Figure 4.3, any block with formal argument fields can use numeric or character string literals as the actual argument. `FillField` represents

[6]This is an example to refine the log data by introducing more context of the action that more accurately describes the learners' intent. With the primitive NetsBlox logging, both Disconnected→Disconnected and Connected→Disconnected would be logged as `setBlockPosition`.

such an action. In addition, `StartSimulation` indicates when a student starts a simulation to test the computational model[7]

Table 4.2: Block state transitions and computational model edit actions

Transition	Action	Description
Not Present → Disconnected	Add	instantiate a new block but do not connect it to anything
Not Present → Connected	Connect	instantiate a new block and connect it to another block
Disconnected → Not Present	Remove	remove/delete a block
Disconnected → Connected	Connect	connect an existing block to another block
Connected → Not Present	Remove	remove/delete a block
Connected → Disconnected	Disconnect	disconnect a block from another existing block
Connected → Connected	Connect	connect an existing block to another block

The following sequence shows the first 20 actions of a student in the computational model-building activity separated by '→':

`AddBlock; ConnectBlock` → `ConnectBlock` → `ConnectBlock` → `FillField` → `DisconnectBlock` → `AddBlock` → `ConnectBlock` → `RemoveBlock` → `AddBlock` → `RemoveBlock` → `AddBlock` → `AddBlock` → `ConnectBlock` → `ConnectBlock` → `ConnectBlock` → `StartSimulation` → `AddBlock` → `RemoveBlock` → `ConnectBlock` → `...` In addition to the type as shown in the sequence, each action also contains metadata such as the `Unix` epoch timestamp, username, project name, sessionId, etc. The action sequence of each student can be aggregated to find distinct patterns that indicate their learning and problem-solving processes (cf. Section 5.4).

4.5.1 Computational Model-building Performance

In addition to deriving behavior measures from students' behaviors, we have developed analytical tools that represent students' computational models as abstract-syntax trees (ASTs) (Bille, 2005) and compare the edit distances to the correct implementations. The method can also transform the trees to semantics-preserving variations (Xu and Chee, 2003) so that the ASTs are *standardized* (e.g, equalizing the syntactically different but semanti-

[7]The engineering design activity also has a start simulation action but they are logged and treated differently to differentiate the type of learning activity, as an extension to the generic `pressStart` action of NetsBlox.

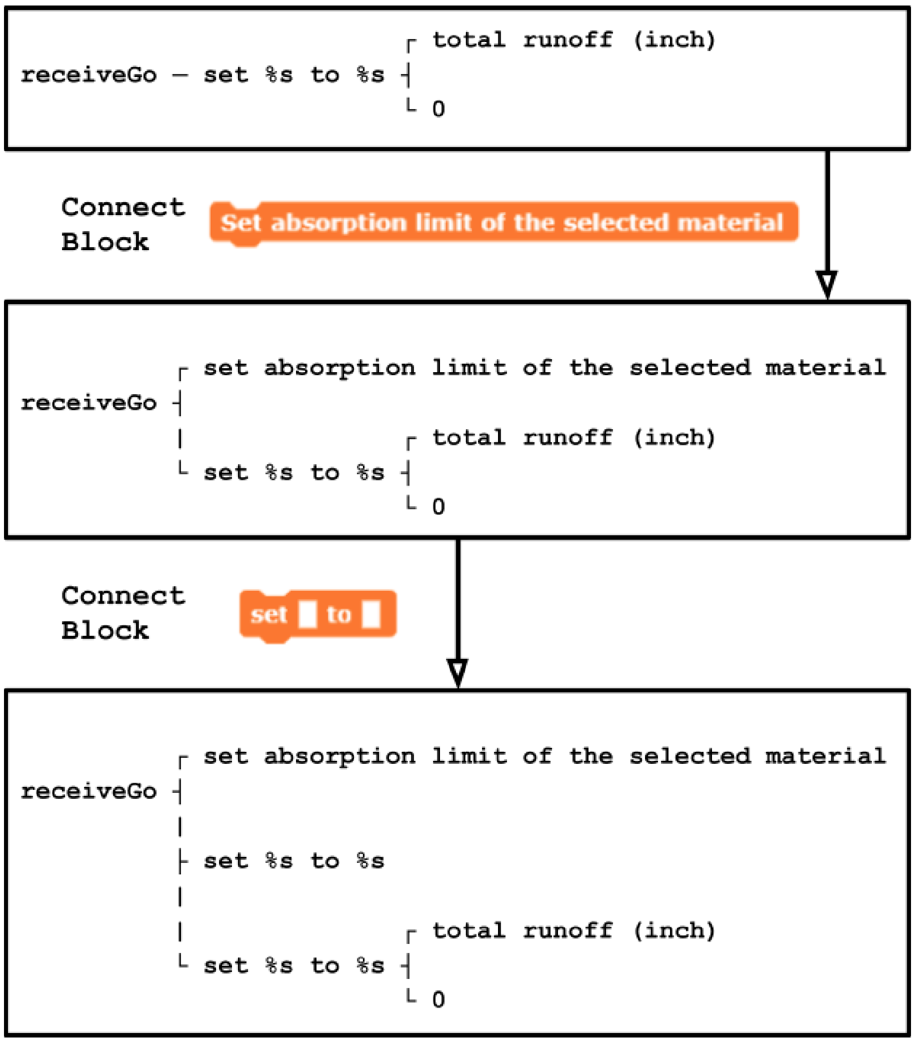
cally isomorphic expressions $a < b$ and $b > a$). Xu and Chee (2003) summarized five types of code transformations of (1) Statement Separation, (2) Temporary Declaration Standardization, (3) Algebraic Expression Standardization, (4) Control Structure Standardization, and (5) Boolean Expression Standardization. The latter 3 types of transformations are more relevant and applied in our learning environment.

```
                                        ┌── total rainfall
     ┌── % is less than % ─┤
     |                                  └── absorption limit
     |
     |                                            ┌── total absorption
IF ──┤          ┌── set % to % ─┤
     └── then ─┤                              └── total rainfall
                |                             ┌── total runoff
                └── set % to % ─┤
                                              └── 0
```

```
if total rainfall < absorption limit then
    total absorption := total rainfall;
    total runoff := 0;
end
```

Figure 4.10: A snippet of the AST of the computational model and its corresponding rule

Figure 4.10 shows a snippet of the AST of the **standardized** computational model reconstructed from the log data. This part of the AST corresponds to the rule that determines the amount of absorption and runoff when the rainfall does not exceed the absorption limit. For *post hoc* analysis, an in-order traversal of the AST could transform it in the same form of the pseudo-code in Algorithm 3 (the right part of Figure 4.10). This method can serve as the basis for measuring the computational model-building performance over time by tracking how the tree-edit distances changed each time when the computational model was modified. Figure 4.11 presents an example sequence of two consecutive computational model edits represented as three ASTs in the early phase of the computational model-building task. In this example, both actions are effective (correct).



```
                            ┌ total runoff (inch)
receiveGo ─ set %s to %s ┤
                            └ 0
```

```
          ┌ set absorption limit of the selected material
receiveGo ┤
          |                  ┌ total runoff (inch)
          └ set %s to %s ┤
                             └ 0
```

```
          ┌ set absorption limit of the selected material
receiveGo ┤
          |
          ├ set %s to %s
          |
          |                  ┌ total runoff (inch)
          └ set %s to %s ┤
                             └ 0
```

Figure 4.11: Three ASTs before and after two consecutive computational model edit actions

4.6 Analyzing Engineering Design Activities

The system measures the quality of students' designs and their learning behaviors during **engineering design** activities as **performance** measurements. The quality measurements for each student are (1) the number of satisfying schoolyard designs, (2) the lowest amount of runoff caused by all satisfying designs, and (3) the design scores of each design (see Figure 4.12).

As presented in Section 4.2, the schoolyard design needs to follow the three criteria of runoff, cost, and accessibility. Because the values associated with the three design criteria had widely different scales, we applied a simple transformation to each criterion to reduce them to a value between 1.0 and 5.0.

Figure 4.12 presents a visualization of a design that meets all of the criteria, i.e., a design costing $750,000, resulting in 1.02 inches of runoff after a 2-inch rainfall, and having 6 accessible squares on the schoolyard. The score computed for this design is 4.25. The students were not aware of this scoring system while they worked on their designs. Instead, they directly compared the cost of the playground and the amount of runoff for a number of designs. The visualization of scores was not accessible from the students as well.

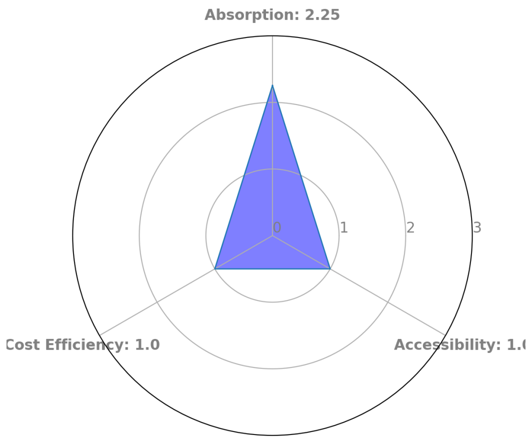


Figure 4.12: Radar chart visualization of the score of a schoolyard design

The number of satisfying schoolyard designs and the lowest amount of runoff caused by satisfying designs can work as additional performance measurements to evaluate the students' performance in the engineering design activity.

4.6.1 Engineering Design Behaviors

Figure 4.13 provides a visualization of a student's *behaviors* during designing and testing his designs projected onto a 3-dimension space. The three axes of the figure correspond to the runoff, cost, and accessibility aspects of the design criteria. Each dot on the 3-D plot marks a tested design. The shaded region stands for a satisfying solution space, and all dots contained in the solution space mark a satisfying design. The 2-D plot (Figure 4.14) presents the same student's design solutions. The dashed lines on the 2-D plot represent the optimal runoff performance for a satisfying solution (0.9625 inches of runoff after 2 inches of rainfall), the cost ($750,000), and accessibility (6 accessible squares) criteria.

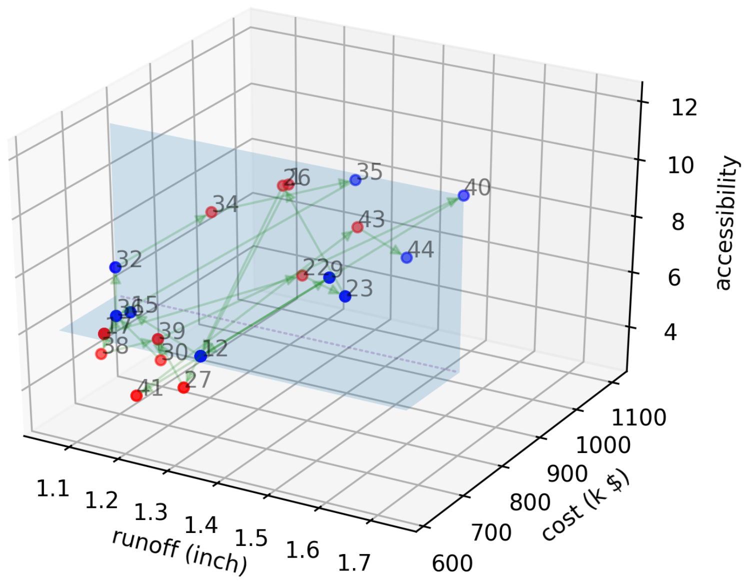


Figure 4.13: 3-D plot visualization of student A's tested engineering designs

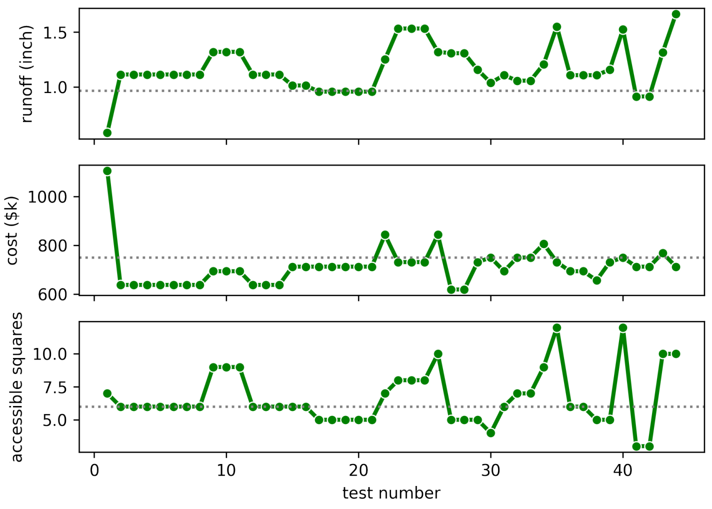


Figure 4.14: Line plot visualization of student A's tested engineering designs

The behavior measurements of the engineering design activity include (1) the number of tests of the designs and (2) the total standardized Euclidean distance (length of the arrow in a 3-D plot) between a student's m designs calculated as

$$\sum_{i=1}^{m-1} \|(V_{i+1} - V_i)^2\| \text{ where } V = \langle runoff_z,\ cost_z,\ accessiblilty_z \rangle$$

The subscript $_z$ indicates the standardized value of runoff, cost, and accessibility of a design after z-transformations because these values are on different scales after z-transformations because these values are on different scales.

The total standardized Euclidean distance and the number of tested designs can be easily calculated with the generation of the 2-D and 3-D design diagrams. These two measures indicate the extent a learner explored the engineering design *experiment space* (Klahr and Dunbar, 1988). In our previous work, we have found that students who explored a larger portion of the problem space were more likely to discover better design solutions in the WRC (Zhang et al., 2019). In this dissertation research, we further explore how students connect the *hypothesis space* and the *experiment space* by making inference with

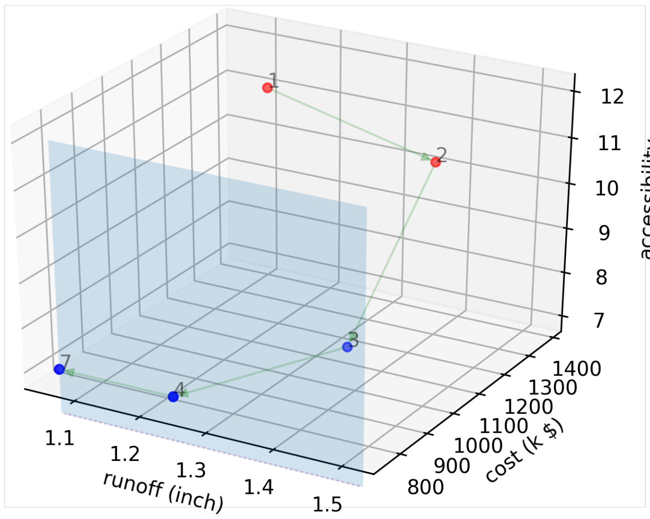


Figure 4.15: 3-D plot visualization of student B's tested engineering designs

data drawn from experiments, according to the Scientific Discovery as Dual Search theory (Klahr and Dunbar, 1988). For example, as shown in Figure 4.15, the total standardized Euclidean distance of Student B's designs is much smaller than Student A's (see Figure 4.13).

4.7 Summary of Chapter

The design and implementation of the computational model-building and engineering design environment are presented and discussed in this chapter to answer the first part of *RQ 1* of this dissertation research, i.e., *How do we design and implement a computational modeling environment to support the integrated learning of science and engineering and how effective is this curriculum in supporting student learning?*

We presented the computational model-building and engineering design learning environment in the WRC, which is one of the first examples of NGSS-aligned curricula that support the interdisciplinary learning of science, engineering, and CT. The learning environment supports the WRC curriculum by enabling computational-modeling activities for students to develop and practice CT instead of performing engineering design with a pre-built model.

As a result of the design and implementation of the DSML, students' learning activities are tailored at an appropriate level of abstraction that fits their computing and mathematics proficiency. In addition, students' interaction with the environment is interactive and engaging, which in turn created an ample amount of context-rich log data. As a result, the behavior and performance measurements calculated from the traces of students' actions in the computational model-building and engineering design activities provide us a holistic view to understand students' learning and problem-solving processes in the challenging learning tasks provided in the WRC curriculum.

For example, the standardized Euclidean distance between designs, dot product, and cosine distance together show the variability and systematicity of a student's engineering design process in addition to their effort exploring the problem space to derive satisfying designs. More specifically, we hypothesize that the students demonstrate good problem-solving behaviors if they first try a wide variety of designs (indicated by large standardized Euclidean distance); and then narrow down their search (indicated by small Euclidean distance) to arrive at a satisfying design.

With the computational model-building and engineering design analyses framework implemented in this dissertation research, we can

1. Retrieve all information of students' edits to their computational models as long as they have logged in.

2. Map the missing or mislabeled user/project/actions with session-ID lookup and fuzzy fingerprint matching so that the trace data are complete.

3. Add a layer of intent(/context)-preserving user actions along with the existing NetsBlox log data. As a result, we can generate action sequences that more accurately reflect what the students performed in the WRC.

4. Build an AST-based data structure to trace each time a model was edited with a very small memory and space overhead. This serves as the basis to determine if an edit is correct by comparing it to canonical implementations of the model.

Next, in Chapter 5, we present a classroom study and the processes to (1) grade the assessment, (2) collect data, and (3) generate the behavior and performance measurements. We also introduce Path Diagram Analysis, Markov Chain modeling, and Sequential Pattern Mining —analytical tools used to generate the results with variables introduced in this chapter.

Some of the methods of this chapter (i.e., the design and implementation of DSML blocks (Figures 4.3 and 4.4), the radar chart visualization of the design scores Figure 4.12, and the 3-D visualization of students' engineering design processes (Figures 4.13, 4.15) have been reported in Zhang et al. (2019); McElhaney et al. (2020), and Zhang et al. (2020).

Chapter 5

Methods

5.1 Study and Procedure

The data reported in this dissertation research was collected from a classroom study in a 6th-grade classroom in the southeastern United States. 99 students participated in the study, which was run over 15 school days in the Fall of 2019. The classroom study was led by two experienced science teachers. Three Vanderbilt researchers provided additional support in the classroom but mostly acted as observers during the study. The two teachers received four days of training from the research team (consisted of Vanderbilt researchers and external collaborators from the University of Virginia and SRI International) in the summer session before the study was conducted in the Fall. The school where the study was conducted offered elementary programming classes as part of its curriculum, and all participating students had varying amounts of prior programming experience with Scratch (https://scratch.mit.edu/about).

During the study implementation, students worked for 45 minutes per day, three days a week in their regular science classes, and 75 minutes, twice a week with additional personalized-learning time. The Water Runoff Challenge (WRC) curriculum was covered in 15 school days, with the pre-post tests administered during two additional 45-minute class sessions. Figure 5.1 shows the learning progression applied in the WRC curriculum (image provided by collaborators from the University of Virginia). Table 5.1 lists a break-down of the themes of the day-by-day learning activities.

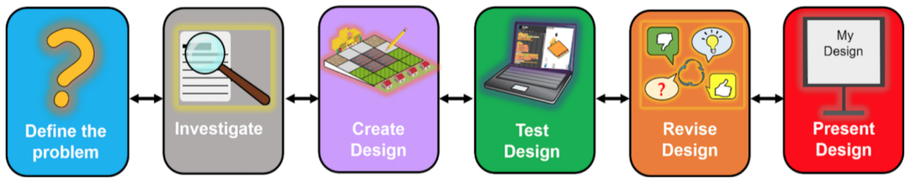


Figure 5.1: Learning progression of the engineering design curriculum

Table 5.1: Themes of the learning activities of the 15-day WRC implementation

lesson	theme
1	What problem are we trying to solve?
2	Where does the rain go?
3	How much does it rain at the school?
4	Why do some materials soak in more water?
5	How do we calculate water runoff?
6	How can we design the schoolyard to reduce water runoff?
7	What language does a computer understand?
8	Build a computer model (part 1)
9	Build a computer model (part 2)
10	Build a computer model (part 3)
11	How can we test if our schoolyard design meets the project criteria?
12	How do you know what design will be the best?
13	How can you use the model to improve your design?
14	How can you convince the Principal of the school to use your design?
15	Class presentations

The WRC unit consists of rich activities that include (1) hands-on activities in which students conduct physical investigations on the absorption of different surface materials (see Figure 5.2); (2) create conceptual modeling of the runoff system using a pictorial representation (see Figure 5.3); and (3) present their methods and final engineering designs (see Figure 5.4).

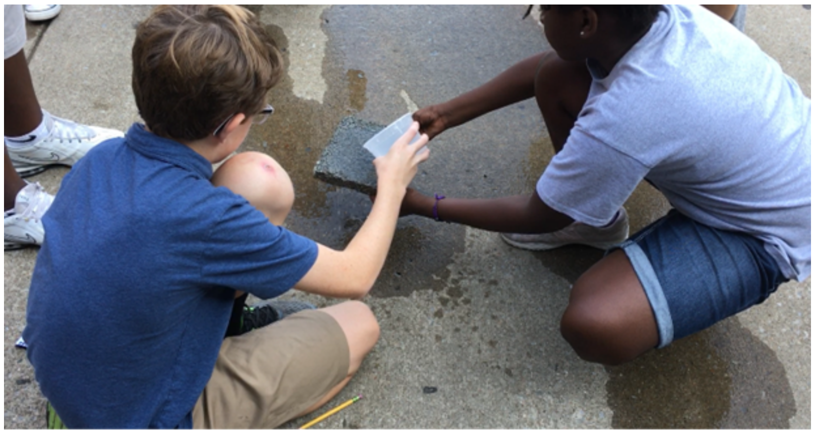


Figure 5.2: Students conducting investigations on the absorbency of different surface materials (image showing the permeable concrete that can be used for a parking lot to reduce standing water and runoff)

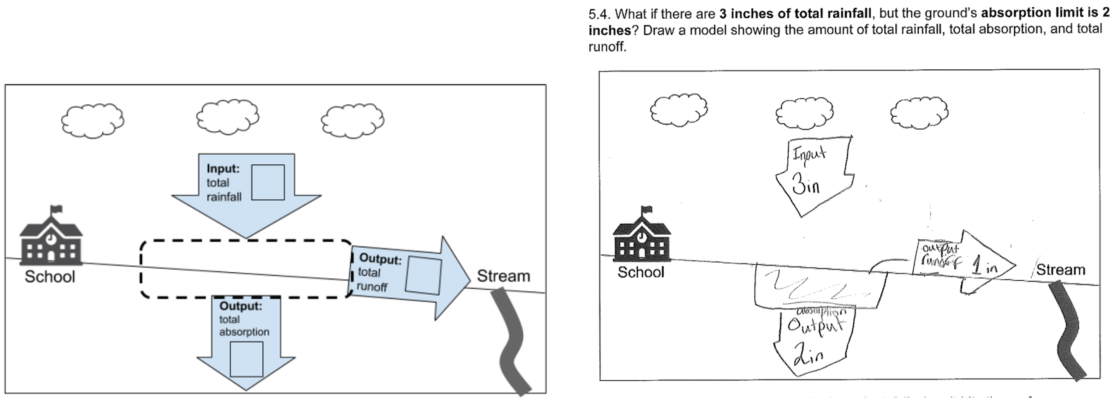
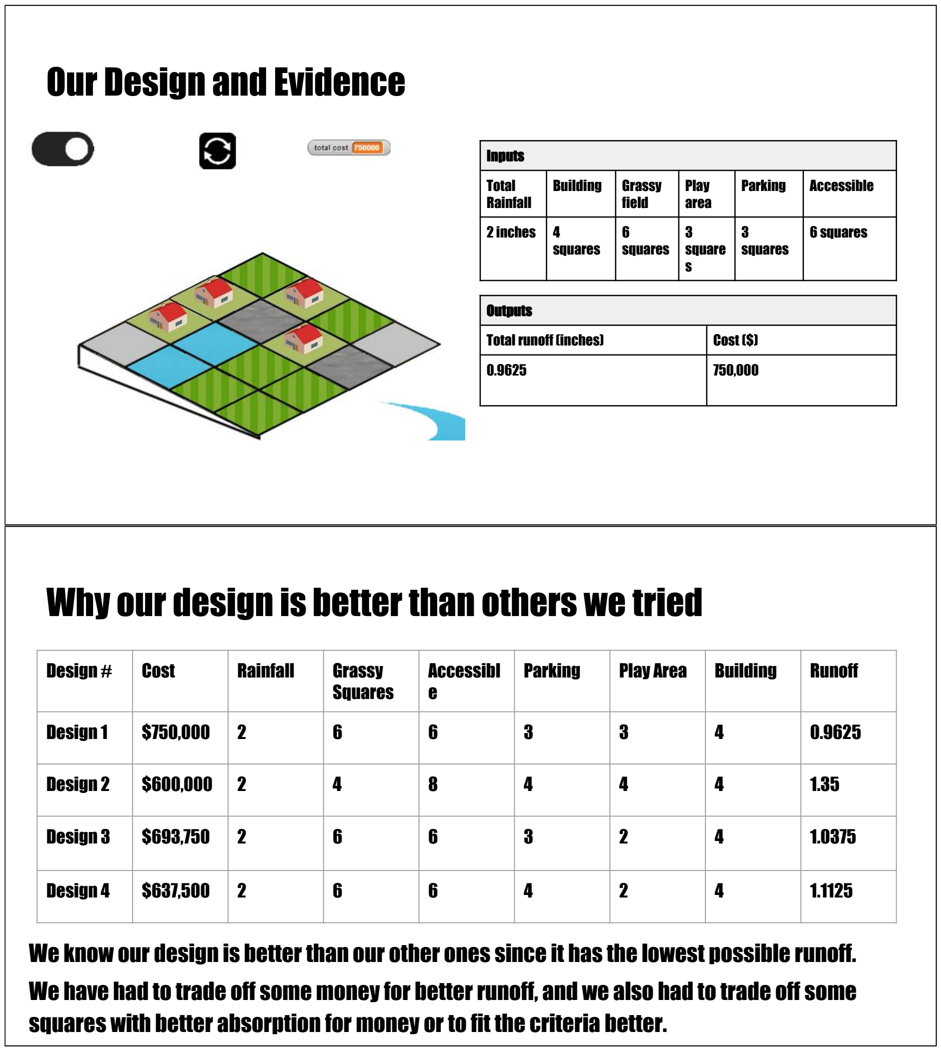

5.4. What if there are **3 inches of total rainfall**, but the ground's **absorption limit is 2 inches**? Draw a model showing the amount of total rainfall, total absorption, and total runoff.

Figure 5.3: A conceptual model of the runoff system (left) and a sample model drawn by a student (right)


Our Design and Evidence

Inputs					
Total Rainfall	Building	Grassy field	Play area	Parking	Accessible
2 inches	4 squares	6 squares	3 squares	3 squares	6 squares

Outputs	
Total runoff (inches)	Cost ($)
0.9625	750,000

Why our design is better than others we tried

Design #	Cost	Rainfall	Grassy Squares	Accessible	Parking	Play Area	Building	Runoff
Design 1	$750,000	2	6	6	3	3	4	0.9625
Design 2	$600,000	2	4	8	4	4	4	1.35
Design 3	$693,750	2	6	6	3	2	4	1.0375
Design 4	$637,500	2	6	6	4	2	4	1.1125

We know our design is better than our other ones since it has the lowest possible runoff. We have had to trade off some money for better runoff, and we also had to trade off some squares with better absorption for money or to fit the criteria better.

Figure 5.4: Sample slides from a student group's final presentation that used a screenshot of their optimal solution and inter-design comparison of satisfying solutions

The data analyzed in this dissertation study consisted of (1) the NGSS-aligned science and engineering + CT pre-post assessments; (2) formative assessments in science, engineering, and CT administered as homework; and (3) system logs of students' model building activities as well as their engineering design and testing activities (lessons 8 - 13). Students, parents, and the two teachers signed permissions that were reviewed and approved by the Vanderbilt Institutional Review Board. We did not miss data from any participant.

5.2 Assessments and Grading

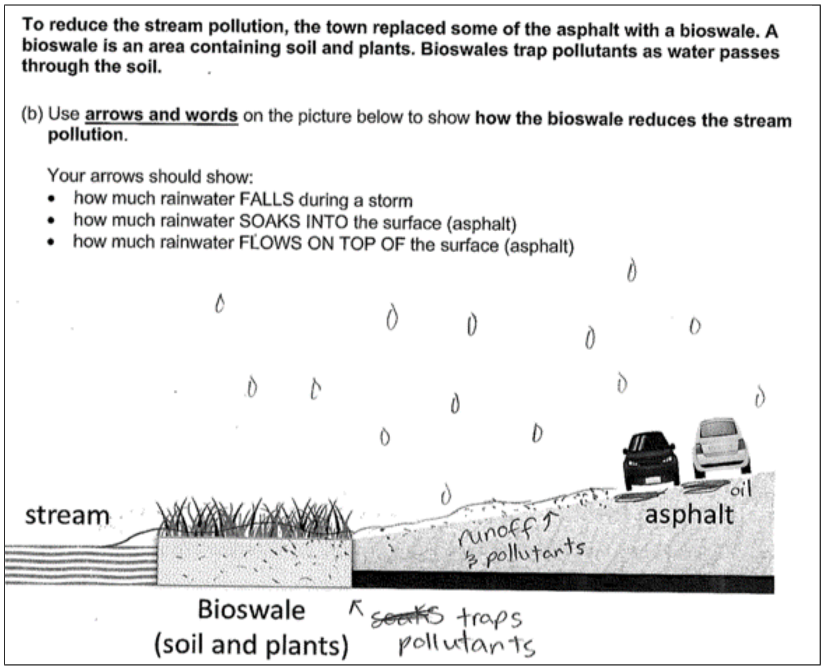
NRC (2010) suggests that the learning goals of engineering be embedded and mapped to the standards of other STEM disciplines. The *summative* science and engineering assessments were designed to align with various NGSS Performance Expectations (PEs) to target the specific science and engineering constructs (McElhaney et al., 2019, 2020). The PEs include, for example, defining a design problem (3-5-ETS1-1) and generate and compare multiple possible solutions (3-5-ETS1-2). The assessment modality included multiple choice and constructed response questions. The CT assessments align with the CT concepts and practices included as part of the modeling activities in the WRC curriculum. Figures 5.5, 5.6, and 5.7 show sample science, engineering, and CT pre-post assessment items using various assessment modalities of drawing, short-answer response, and multiple-choice questions. The complete summative assessment and its rubrics are included in Appendix A of this dissertation.

For the sample science item (Figure 5.5), students needed to draw a pictorial representation that indicated:

1. the precipitation process (falling rainwater);
2. part of rainfall being absorbed into the concrete surface;
3. the remaining portion of rainfall becoming runoff and flowing down the slope;
4. the runoff carrying pollutants downward;
5. the amount of absorption less than the amount of runoff (for the parking lot, optional);

6. the bioswale absorbing water;
7. the remaining runoff flowing through the bioswale;
8. the bioswale removing some pollutants;
9. the amount of absorption more than the amount of runoff (for the bioswale, optional);

On the other hand, factors such as the aesthetics, the evaporation process, and other human activities (e.g., people walking on the slippery slope) are not scored or considered in this item.


To reduce the stream pollution, the town replaced some of the asphalt with a bioswale. A bioswale is an area containing soil and plants. Bioswales trap pollutants as water passes through the soil.

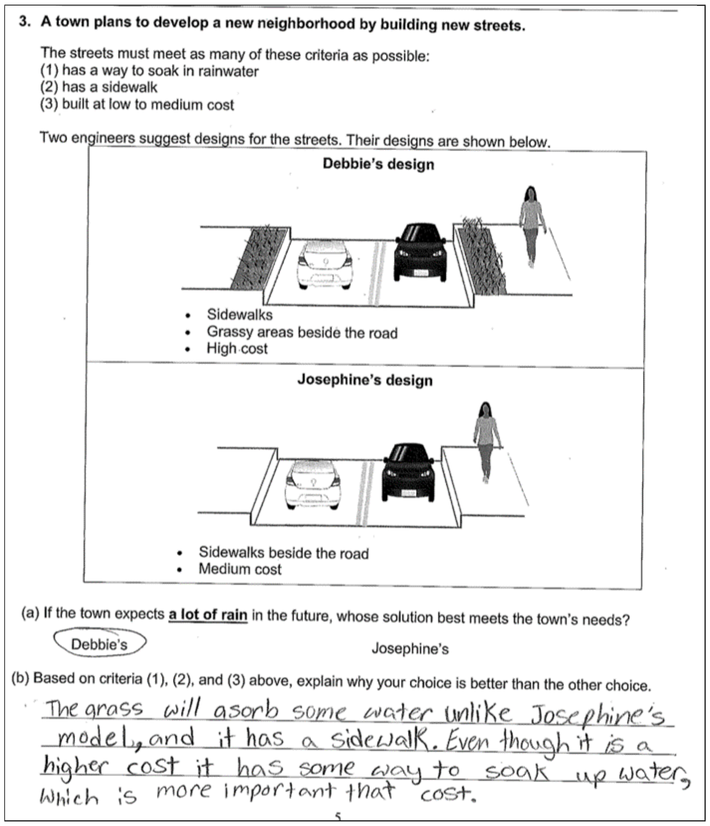
(b) Use **arrows and words** on the picture below to show **how the bioswale reduces the stream pollution.**

Your arrows should show:
- how much rainwater FALLS during a storm
- how much rainwater SOAKS INTO the surface (asphalt)
- how much rainwater FLOWS ON TOP OF the surface (asphalt)

Figure 5.5: A sample science assessment item of the WRC curriculum (with student response)

For the sample engineering test item (Figure 5.6), students needed to trade off the cost and utility between two sub-optimal design solutions that did not meet all of the design criteria. More specifically, in part b of this item, the students needed to indicate by a written response that

1. the first design had a grassy area beside the roads that can absorb some rainwater;
2. the second design did not have a way to reduce runoff;
3. the first design failed to meet the cost criterion;
4. neither design was satisfactory (optional);



3. A town plans to develop a new neighborhood by building new streets.

The streets must meet as many of these criteria as possible:
(1) has a way to soak in rainwater
(2) has a sidewalk
(3) built at low to medium cost

Two engineers suggest designs for the streets. Their designs are shown below.

Debbie's design

- Sidewalks
- Grassy areas beside the road
- High cost

Josephine's design

- Sidewalks beside the road
- Medium cost

(a) If the town expects **a lot of rain** in the future, whose solution best meets the town's needs?

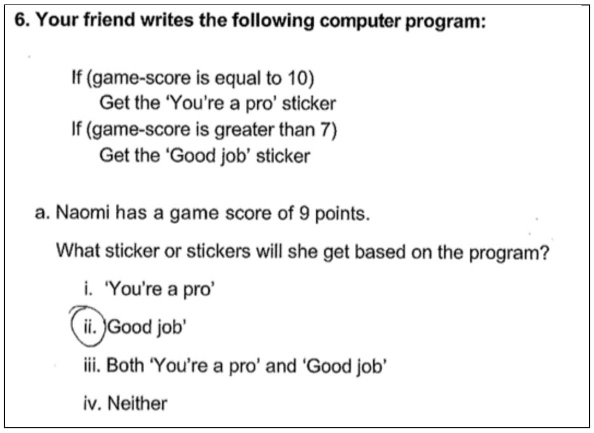
[✓ Debbie's] Josephine's

(b) Based on criteria (1), (2), and (3) above, explain why your choice is better than the other choice.

The grass will asorb some water unlike Josephine's model, and it has a sidewalK. Even though it is a higher cost it has some way to soak up water, Which is more important that cost.

Figure 5.6: Sample engineering assessment items of the WRC curriculum (with student response)

The sample CT item (Figure 5.7) evaluates students understanding of conditional logic and has been used in previous research studies (e.g., (Basu et al., 2016a, 2017; Zhang et al., 2017).



6. Your friend writes the following computer program:

```
If (game-score is equal to 10)
    Get the 'You're a pro' sticker
If (game-score is greater than 7)
    Get the 'Good job' sticker
```

a. Naomi has a game score of 9 points.

What sticker or stickers will she get based on the program?

i. 'You're a pro'

[✓ ii. 'Good job']

iii. Both 'You're a pro' and 'Good job'

iv. Neither

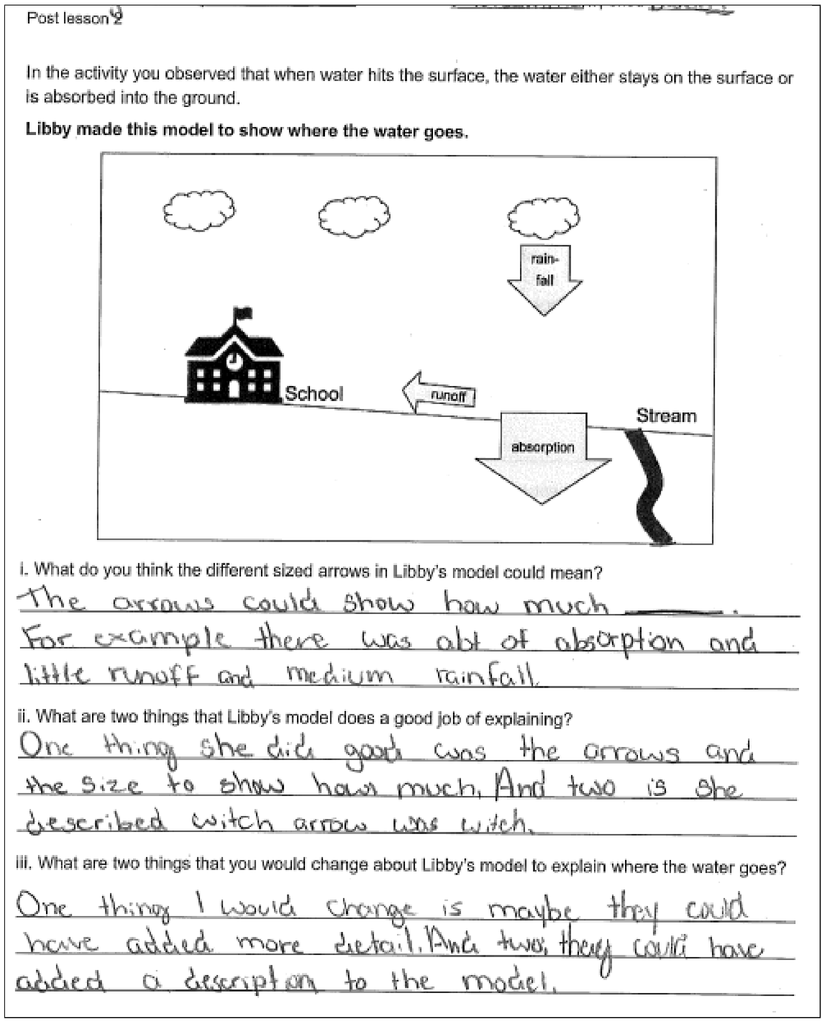
Figure 5.7: Sample CT assessment items of the WRC curriculum (with student response)

Rubrics for coding and scoring the pre-post assessments were refined from McElhaney et al. (2019). The rubrics rewarded the extent to which students could explain the causes and effects of the scientific concepts of water absorption and runoff, as well as make valid engineering decisions informed by the underlying scientific reasoning (Zhang et al., 2019). More detailed discussions of the development of the 3-dimensional assessment, its alignment to the NGSS, and the description of the grading rubrics have been presented in (Chiu et al., 2019; McElhaney et al., 2019).

Before grading, two researchers received five hours of training on the rubrics, during which 5% of test submissions were randomly selected and graded together. Another 20% test submissions were then individually graded by the same researchers to establish inter-rater reliability (Cohen's κ at ≥ 0.8 level on all items). All differences in coding between the two researchers were discussed and resolved before the remaining 75% of test submissions were graded by one of the researchers.

The *formative* assessment tasks were designed to closely mirror various curricular activities. For example, science assessments measured students' understanding of the water conservation relations and the relative effect of different surface materials on water runoff, while the engineering tasks measured students' ability to compare different solutions and make appropriate trade-offs. The CT assessments measured students' ability to comprehend incorrect code to predict water runoff and absorption, and debug such code. There are 14 tasks in the formative assessment. These tasks were scored using a simple correct/incorrect rubric with 31 possible points in total.[1]

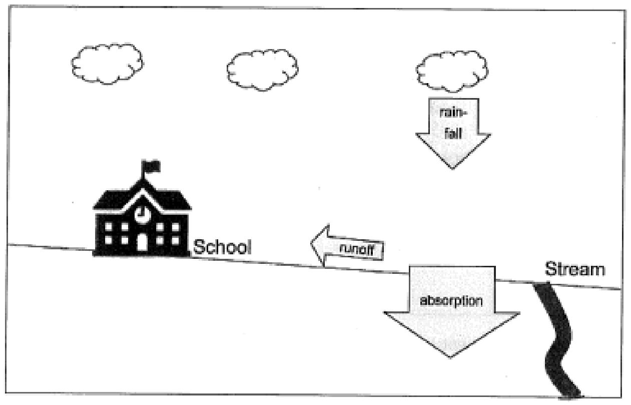
[1]Collaborators at SRI International are the major contributors that created the assessments and rubrics.



Post lesson 2

In the activity you observed that when water hits the surface, the water either stays on the surface or is absorbed into the ground.

Libby made this model to show where the water goes.



i. What do you think the different sized arrows in Libby's model could mean?

The arrows could show how much. For example there was alot of absorption and little runoff and medium rainfall.

ii. What are two things that Libby's model does a good job of explaining?

One thing she did good was the arrows and the size to show how much. And two is she described witch arrow was witch.

iii. What are two things that you would change about Libby's model to explain where the water goes?

One thing I would change is maybe they could have added more detail. And two, they could have added a description to the model.

Figure 5.8: Sample formative assessment items of the WRC curriculum (with student response)

5.3 Path Analysis

Path Analysis extends linear regression and multiple regression methods. The traditional regression methods assume that (1) only direct effects exist between dependent and independent variables and (2) errors in the dependent variable are uncorrelated with the independent variable (Ahn, 2002; Wright, 1983). On the other hand, when applied to study intrinsically related variables where indirect variables play a mitigating role, multi-regression or correlation analysis does not provide optimal model estimates (Pearl and Mackenzie, 2018). Path Analysis predates and can be seen as a variation of Structural

Equation Modeling (Kline, 2015) without the latent variables. In this work, we use Path Analysis to study the effects and the relative importance of effects among the measured performance and behavior values.

In particular, we used path analysis to study the relationship between science, engineering, and CT learning in the WRC. We hypothesized that students' knowledge, behaviors, and performances could influence their subsequent learning behaviors, performances, and summative test scores in the WRC curriculum. Conceptually, we hypothesized causal paths as shown in Figure 5.9. Each arrow in the diagram indicates a direct effect on the endogenous variable from the exogenous variable. The horizontal positions of the variables also correspond to their temporal order in the WRC curriculum (pre-tests $\prec$ formative assessments $\leq$ computational modeling $\prec$ engineering design $\prec$ post-tests).

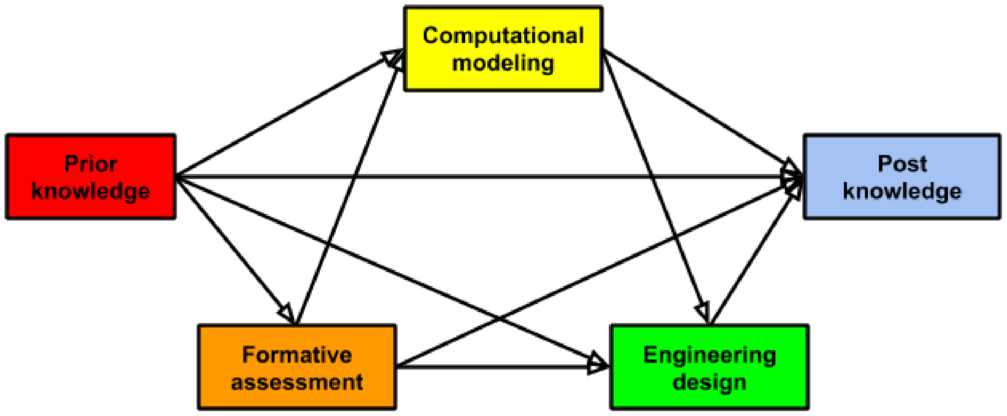


Figure 5.9: A hypothesized path model of the direct effects on the different categories of learning behavior and performance variables

5.4 Pattern Mining Using Markov Chain Modeling

In Chapter 3, we defined students' learning strategies as *students' conscious and controllable action sequences to facilitate and enhance task performance.* We also assume that instead of being random or reactive responses, students' behaviors in a learning activity reflect their use of learning strategies. Therefore, the likelihood of performing certain sequences of actions can help us infer the learning strategies they apply to their model building, engineering design, and problem-solving tasks. Students' strategies can vary substantially, and these differences are indicative of differences in their model building performance and learning (Kinnebrew et al., 2014; Kinnebrew et al., 2017).

When a student engages in the WRC, his or her learning activity can be represented as a sequence of actions generated from the log data. Therefore, the probability of a state transition among the actions defines a *Markov chain* (MC) on the state space (Russell and Norvig, 2003).

For example, an arbitrary action at an arbitrary time can be represented as a state, and the occurrence of another action following the first action creates a state transition from the first action (state) to the second. In addition, a transition, e.g.,

$$\texttt{AddBlock} \xrightarrow{\quad 0.54 \quad} \texttt{ConnectBlock}$$

is associated with a conditional probability, i.e.,

$$\Pr(action_{t+1} = \text{ConnectBlock} \mid action_t = \text{AddBlock}) = \frac{\#\text{AddBlock} \rightarrow \text{ConnectBlock}}{\#\text{AddBlock}} = 0.54$$

Taking into account the learning activities of a group of students working on the WRC, we can derive an MC model that represents and predicts the students' dynamic learning processes. For example, Figure 7.1 provides a visualization of the aggregated first-order Markov Chain model of the computational model-building activities of all participants in the study ($N = 99$). In addition, the MC model can also estimate the likelihood of transitions given sequences of states in the state space.

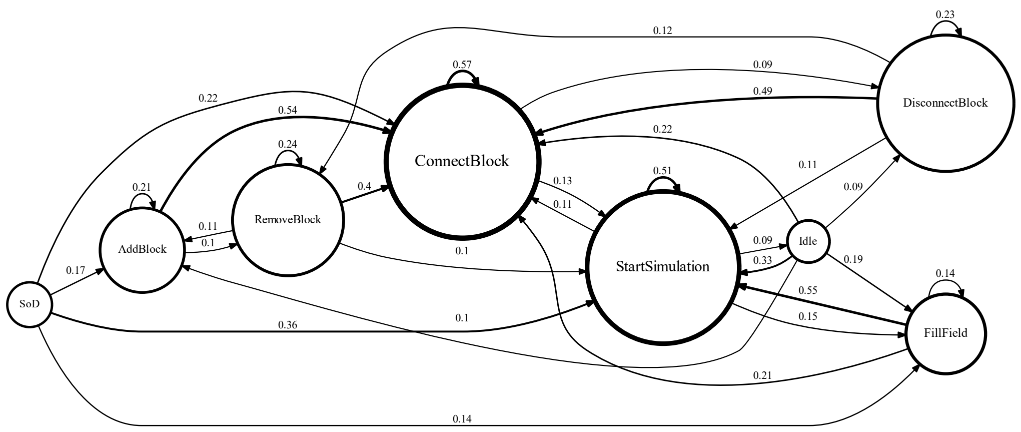


Figure 5.10: Sample Markov chain model of students' computational model-building processes

Furthermore, with the likelihoods estimated from groups of students (e.g., high versus low performing), we can perform hypothesis tests to check if the distributions of the likelihoods are statistically significant using methods like the Mann-Whitney U-test (Siegal, 1956). Algorithm 4 illustrates the process of generating likelihood distributions and conducting hypothesis tests.

Additionally, we applied a lift measurement (Merceron and Yacef, 2008) that defines the *interestingness* and importance of an association rule consisting of the *rule body* (an action or action sequence A) and *rule head* (an action B). The value of the lift is the odds of the confidence of the rule and the expected confidence of the rule calculated as

$$lift = \frac{Confidence(A \rightarrow B)}{P(B)} = \frac{P(A \wedge B)}{P(A)P(B)}$$

Therefore, the lift a measure reflects the deviation of the rule from the model of statistic independence of the rule body and rule head. The measurement takes a value between 0 and inf. Patterns or rules having a lift > 1.0 implies its relatively higher importance because it indicates that the rule body and the rule head appear more often together than expected. Therefore, the occurrence of the rule body has a positive effect on the occurrence of the rule head (Merceron and Yacef, 2008).

On the other hand, a lift measure smaller than 1.0 indicates that the rule body and the rule head appear less often together than expected, this means that the occurrence of the rule body has a negative effect on the occurrence of the rule head. Therefore, association rules having very small lifts are also interesting.

5.5 Summary of Chapter

This Chapter presents the setting and procedure of the classroom study, as well as the design and grading of the NGSS-aligned science and engineering and CT assessments. Two analytical approaches, i.e., Path Analysis (Wright, 1983; Pearl and Mackenzie, 2018) and Markov Chain Modeling (Russell and Norvig, 2003) are proposed to study (1) the associations and impacts of the learning activities in the WRC that are evaluated by a series of behavioral and performance measurements and (2) students' application of learning strategies during these learning activities.

Using the measurements described in Chapter 4 and the methods referenced in this Chapter, we present, analyze, and discuss the students' learning outcomes in the WRC in Chapter 6 and the students' use of learning strategies in Chapter 7.

Algorithm 4: Discovering action sequences

```
Result: Statistically significant sequences
smoothing_term := $\frac{1}{2^{16}}$ ;
performance := Map(student: score) ;
median := median(scores) ;
high_group_models, low_group_models := Array(), Array();
for student ∈ students do
    for action ∈ actions do
        count instances of action;
        if action_count == 0 then
            prior_probability := smoothing_term;
        else
            prior_probability := action_count / number_of_actions;
        end
    end
    for action_pair ∈ action_pairs do
        count instances of action_pair;
        if pair_count == 0 then
            conditional_probability := smoothing_term$^2$;
        else
            conditional_probability := pair_count / action_count;
        end
    end
    mc := MarkovChain(prior_probability, conditional_probability);
    if performance[student] > median then
        high_group_models.add(mc);
    else
        low_group_models.add(mc);
    end
end
for sequence ∈ action_sequences do
    high_group_likelihoods, low_group_likelihoods:= Array(), Array();
    for mc ∈ high_group_models do
        high_group_likelihoods.add(mc.getLikelihood(sequence));
    end
    for mc ∈ low_group_models do
        low_group_likelihoods.add(mc.getLikelihood(sequence));
    end
    hypothesis := hypothesisTest(high_group_likelihoods,
      low_group_likelihoods);
    if hypothesis.p-value < 0.05 then
        result.add((sequence, hypothesis));
    end
end
```

Chapter 6

Learning Outcomes in the WRC Curriculum

This chapter investigates three research questions of this dissertation, i.e., *How do we design and implement a computational modeling environment to support the integrated learning of science and engineering and how effective is this curriculum in supporting student learning? (RQ 1)*; *What are the relationships between science learning and engineering performance? (RQ 2)*; and *What is the role of computational thinking in facilitating science learning and engineering design? (RQ 3)*. It reports the students' learning outcomes in the WRC curriculum described in Chapter 5, including (1) the summative domain and CT pre-post test, (2) the integrated formative assessment, (3) the behaviors and performances in the computational modeling activity, and (4) the behaviors and performances in the engineering design activity.

To investigate *RQ 2* and *RQ 3*, we used hypothesis testing, the analysis of variance (ANOVA), the analysis of covariance (ANCOVA), correlation analysis, and Path Analysis (Lowry, 2014; Kokoska and Zwillinger, 2000; Wright, 1983) on variables to measure student learning (c.f. Sections 3.2, 4.5 4.6, and 5.2). Table 6.1 summarizes the operationalized variables measuring the learning outcomes.

6.1 Domain and CT Learning

Students' pre-post test scores were compared to determine their learning gains in science, engineering, and CT. Figure 6.1 visualizes the distribution of the test scores. To check the normality of the scores, we first measured the skewness (z-value=-0.811, p-value=0.417) and kurtosis (z-value=-0.567, p-value=0.571) of the score distributions and confirmed that they were close to a normal distribution. Table 6.2 presents the normality test statistics. Both the skewness and the kurtosis (see Table 6.2) satisfied the requirement for a normal distribution (Kokoska and Zwillinger, 2000).

Table 6.1: Description of the measured variables

	variable	domain	description
1	pre sci	science	prior knowledge of science
2	pre eng	engineering	prior knowledge of engineering design
3	pre ct	CT	prior knowledge of CT
4	formative	ALL	performance in the formative assessment
5	comp edit	CT	the number of edits of the computational model
6	comp test	CT	the number of tests of the computational model
7	edit btw test	CT	the median number of edits between two computational model tests
8	comp model score	science, CT	the score of the computational model
9	engineering test	engineering	the number of tests of schoolyard designs
10	engineering euclid	engineering	the total Euclidean distance of designs projected to the 3-D problem space
11	num satisfy	science, engineering	the number of unique designs that satisfy all criteria
12	lowest runoff	science, engineering	the lowest amount of runoff caused by all satisfying designs
13	post sci	science	post knowledge of science
14	post eng	engineering	post knowledge of engineering design
15	post ct	CT	post knowledge of CT

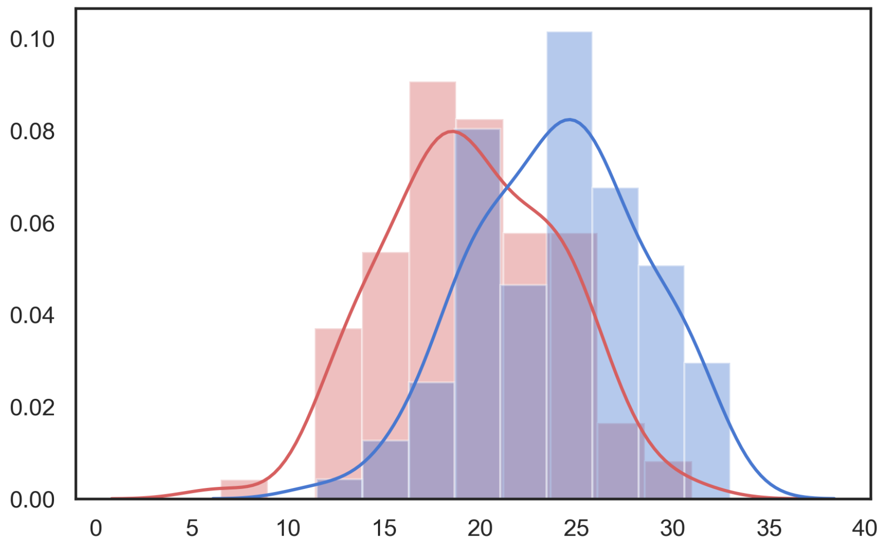


Figure 6.1: Distribution plots (with kernel density estimation) of students' overall pre-post test scores – pre-test: *M* = 19.52, *SD* = 4.47; post-test: *M* = 24.03, *SD* = 4.39

Table 6.2: Normality tests statistics of the summative assessments

Test Statistics		pre-test	post-test
Skewness	population skewness (unbiased)	-0.014	-0.192
	standard error	0.243	0.243
	asymptotic z-value	-0.058	0.243
	p-value	0.954	0.243
Kurtosis	population excess kurtosis (unbiased)	-0.182	-0.301
	standard error	0.481	0.481
	asymptotic z-value	-0.244	-0.567
	p-value	0.807	0.571

Further, we conducted Kolmogorov-Smirnov tests on the assessment scores and found that for the pre-test, the two-sided Kolmogorov-Smirnov statistic (the maximum absolute difference) of D(99) = 0.066 and Lilliefors p-value > 0.1 (Lilliefors, 1967). For the post-test, the two-sided Kolmogorov-Smirnov statistic (the maximum absolute difference) was D(99) = 0.048, Lilliefors p-value > 0.1. Since the p-values are both greater than 0.05, there is not sufficient evidence to reject the hypothesis that the data is normal. Therefore, we used the paired t-test to measure the statistical significance of the difference in the pre-post scores. As shown in Table 6.3, all differences are statistically significant with moderate (≥ 0.5) to large (≥ 0.8) effect sizes.

Table 6.3: Students' ($n = 99$) learning gains and effect sizes

	Total points	Pre-score(*SD*)	Post-score(*SD*)	*p*-value	Cohen's *d*
Science	7	4.56 (1.03)	5.13 (1.04)	<0.001	0.54
Engineering	16	8.73 (2.62)	10.50 (2.67)	<0.0001	0.67
CT	13	6.23 (2.60)	8.41 (2.69)	<0.0001	0.83
Overall	36	19.52 (4.47)	24.03 (4.39)	<0.0001	1.02

To check the influence of the teachers in students' learning gains using the WRC curriculum, we also conducted an one-way ANOVA test (Lowry, 2014) and found that there was no teachers' effect on the normalized change (NC) of the students' learning gains. The one-way ANOVA test results ($F(1,97) = 0.223,\ p = 0.64,\ \eta_p^2 = 0.002$) indicate that the two participating teachers delivered the WRC curriculum at equally high levels.

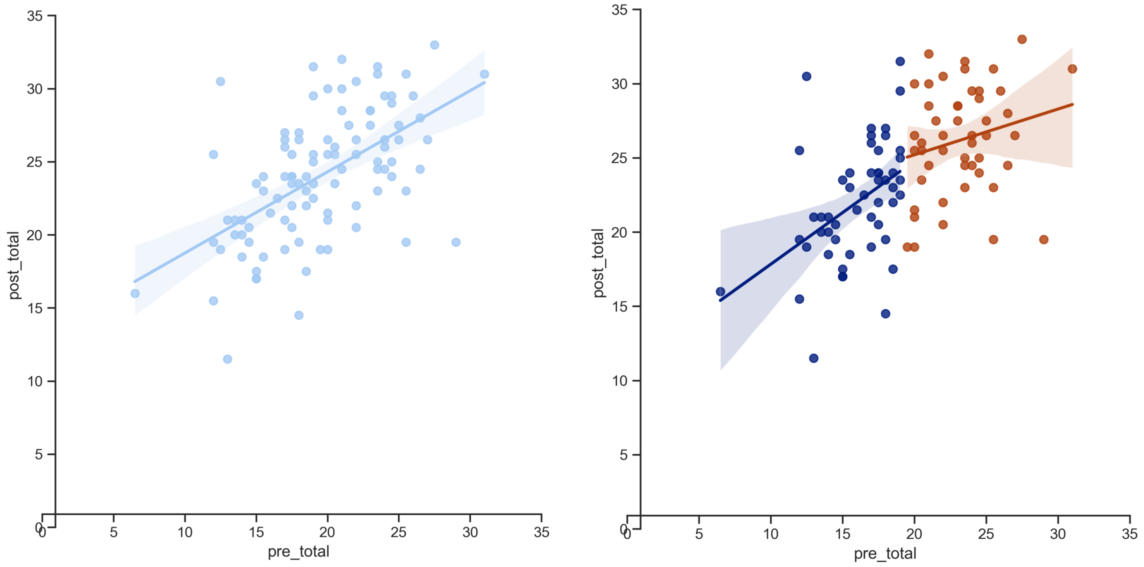


Figure 6.2: Scatter plots of students' learning gains over pre-test assessment scores with regression lines and 95% confidence intervals; left: all students; right: *ad hoc* allocation of students according to pre-test score

To check the effect of prior knowledge in the learning gains, we performed an one-way ANCOVA (Lowry, 2014) with the post-test scores as the dependent variable, an *ad* *hoc* assignment according to the pre-test scores using a median cut (high prior knowledge (HPK)) as the independent variable, and the teacher as the covariate.

Figure 6.2 shows the scatter distribution of the post-test scores relative to the pre-test scores. The figure on the left plots the post- versus pre-test scores for all students, and the plot on the right separates students who started with low prior knowledge from students who had high prior knowledge. The HPK (blue cross marks) and LKP (red circle marks) students were divided and their perspective linear regression lines with 95% confidence intervals are shown in the figure. The difference in the slope of the regression lines suggests that the LKP students had gained more than the HPK students from the intervention. This result is positive because the intervention helped the low-prior-knowledge students to gain more and come closer in performance to the high-prior-knowledge students.

Table 6.4 provides a break-down of scores for high- and low-performing students[1]. It clearly shows that 12 low performing students moved into the high-performing category. However, a majority of the low performers remained low performers overall, which indicates that the intervention needs to be improved to help more low performers. We discuss this in Chapter 8.

Table 6.4: Confusion matrix of qualitative change

		post-test	
		high	low
pre-test	high	36	12
	low	12	39

Results shown in Table 6.5 suggests that there is a significant effect of students' prior knowledge: $F(1,96) = 30.66$, $p < 0.0001$, $\eta_p^2 = 0.233$. This result suggests that while controlling the teacher factors, there is a significant difference in the normalized learning gains caused by the prior knowledge as measured by the pre-test scores given the students relatively high pre-test scores (c.f. Figure 6.1) and the challenging test items. As a result,

[1]The median scores of the pre-test and the post-test used to divide the students were 20 and 24 points, respectively.

the students having less prior knowledge had relatively larger learning gains as well. On the other hand, at least 76% of the variance in the post-test scores cannot be explained by the pre-test scores ($\eta_p^2 = 0.233$), and, therefore, can be attributed to the intervention. The results from the Path Analysis reported in Section 6.5 provides more details on the influence of the curricular activities on the assessment scores.

Table 6.5: Test statistics of one-way ANCOVA to detect the difference among students having higher vs. lower pre-test scores

	Source	SS	DF	F	p-value
0	hpk	455.1	1	30.66	< 0.00001
1	teacher	21.8	1	1.47	0.229
2	residual	1424.9	96	NaN	NaN

6.1.1 Synergistic learning

To study synergy between the learning of science, engineering, and CT, we checked the correlations between the test scores as well as the normalized learning gains (normalized changes). The calculated Spearman's ρ values (Kokoska and Zwillinger, 2000) are reported in Table 6.6. In these calculations, we avoided the impact of outliers and non-normally distributed test scores. Statistically significant correlations ($p < 0.05$) are marked in bold font.

Table 6.6: Spearman's ρ's of the summative assessment scores (statistically significant correlation coefficients ($p < 0.05$) marked in bold font)

	pre sci	pre eng	pre ct	post sci	post eng	post ct	nc sci	nc eng
pre sci	-							
pre eng	-0.03	-						
pre ct	0.13	**0.38**	-					
post sci	0.02	-0.07	-0.08	-				
post eng	-0.07	**0.50**	0.10	0.10	-			
post ct	0.09	**0.43**	**0.62**	-0.16	**0.27**	-		
nc sci	**-0.51**	-0.06	-0.14	**0.82**	0.09	-0.15	-	
nc eng	0.01	**-0.27**	-0.14	**0.20**	**0.62**	-0.06	0.14	-
nc ct	-0.04	**0.22**	0.02	-0.06	**0.24**	**0.73**	0.02	0.06

To begin with, engineering and CT pre-post tests are significantly correlated (ρ = 0.5 and 0.62, respectively). Because of the way the normalized changes (learning gains) are defined, it is not surprising to see high correlation coefficients between the learning gains and the post-test scores (ρ's = 0.82, 0.62, and 0.73). Additionally, the normalized changes in the science and engineering tests are negatively correlated with the pre-test scores (with ρ of -0.51 and -0.27, respectively). It is common and expected that the pre-test scores, post-test scores, and learning gains to be correlated in the context of the pre-post setting of the present study.

Secondly, the normalized changes (learning gains) are not correlated to any other variables. The learning gains in science and engineering pre-post tests (*nc sci* and *nc eng*) had a weak correlation (ρ = 0.14) but is not significant. This result indicates that correlation-based methods do not capture the interrelated relationships between the content subjects.

Thirdly, the significant correlation between *post sci* and *nc eng* (ρ = 0.2) and between *post eng* and *nc ct* (ρ = 0.24) suggest promising results in the synergistic learning between the subjects, especially given that *pre sci* are not correlated with any other variables except *nc sci*.

6.2 Formative Assessments

The average score of the integrated science, engineering, and CT formative assessment (c.f. Section 5.2) was 19.05 points (*SD*= 4.57) out of maximum possible scores of 31 points (10 points for science, 11 points for engineering, and 10 points for CT). A considerable number of students got 10/10 points in the science component, and many had high scores in the engineering and CT components. Figure 6.3 provides a visualization of the formative assessment scores. Table 6.7 presents the descriptive statistics of these scores. In addition, Table 6.8 shows that the students' scores in the three types of formative were significantly correlated with each other.

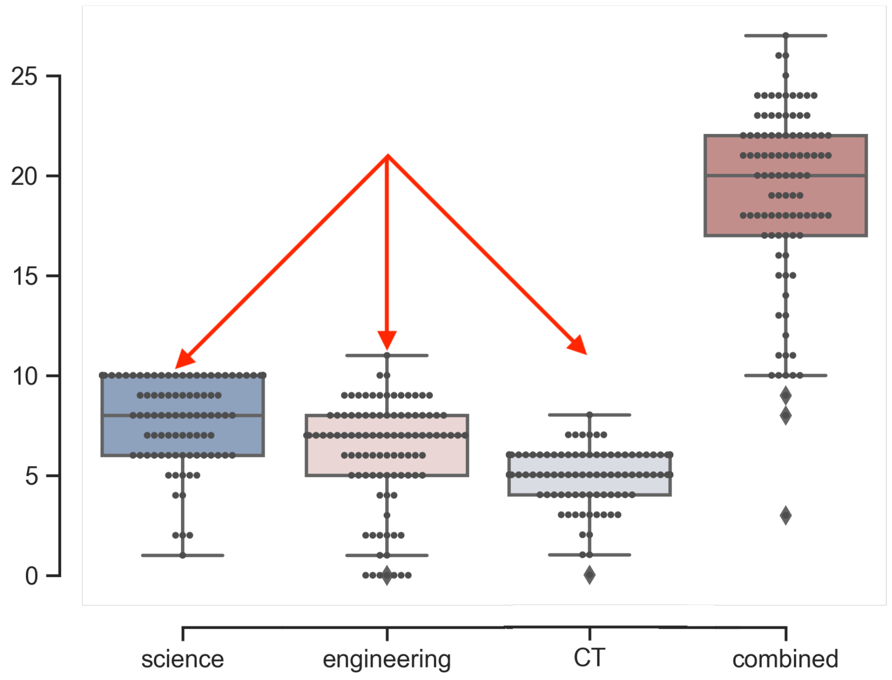


Figure 6.3: Box plots (with marked data points) of students' scores in the formative assessment (arrow points at the maximal scores)

Table 6.7: Descriptive statistics of students' formative assessments

	science	engineering	CT	combined
mean	7.96	6.14	4.95	19.05
std	2.17	2.66	1.39	4.57
min	1	0	0	3
25%	6	5	4	17
50%	8	7	5	20
75%	10	8	6	22
max	10	11	8	27

Table 6.8: Correlation coefficients of the formative assessment scores and learning gains (**: $p < 0.05$, ***: $p < 0.0001$)

	science	engineering	CT
science	-		
engineering	0.23^{**}	-	
CT	0.56^{***}	0.25^{**}	-
pre-post gain	0.00	-0.10	-0.08

These results, along with the pre-post test gains, indicate that the students were learning the domain content, CT concepts and skills, and engineering design practices through the implementation of the WRC curriculum. In addition, the correlations between the formative assessments suggest that these tests were assessing an integrated learning construct (c.f. Section 5.2). On the other hand, none of the formative assessment scores were significantly correlated with the pre-post learning gains, which indicates that the majority of the learning gains contributed from the computational model-building and engineering design activities. For the succeeding analyses, students' combined scores in the science, engineering, and CT categories will be used especially in Section 6.5.

6.3 Computational Model-building Performances

The students showed a large variation in their modeling behaviors. Table 6.9 summarizes the descriptive statistics of the variables measured in the computational model-building activity. On average, they made 167 edits (*SD* = 77) to build their computational models, and they performed 43 tests (*SD* = 47) on them. The average of the median number of edits between tests was 1.11, indicating the student mostly made edits in small chunks between successive model tests.

Table 6.9: Descriptive statistics of the variables in the computational model-building activity

	computational model edits	computational model tests	computational model score
mean	432.4	31.9	4.67
std	211.4	20.5	1.85
min	110	0	0
25%	271	18	4
50%	376	27	6
75%	546	40.5	6
max	934	144	6

Using the method described in Section 4.5.1, we generated abstract syntax tree (AST) representations of students' computational runoff models and then compared them to a series of canonical implementations that are semantically equivalent solutions. AST edit distances were calculated using the Zhang-Shasha algorithm (Bille, 2005). The lowest edit distance between the student model and the canonical implementations were used to calculate the computational model scores for each sub-component of the model in the following way: if the minimal edit distance is 0, the model gets 1 point on that component; if the minimal edit distance is $|\text{Tree}_{student}| + |\text{Tree}_{canonical}|$[2], the model gets 0 points on that component; tree edit distances in between receive a score between 0 and 1 scaled linearly. Because there are three conditions of the runoff scenario and each scenario involves two output variables (c.f. Figure 4.3), the maximum computational model score is 6.0.

In this learning activity, the average computational model score was 4.67 (*SD* = 1.85), and 59% of the students created a correct computational model before the correct model was disclosed in class. Table 6.10 shows the distribution statistics of the scores of the sub-components of the runoff models. The model component with the least number of correct implementations (n = 67) was "`set` *total runoff* `to` (*total rainfall* − *absorption limit*)"

[2]In this case, the two ASTs are completely disjoint and it takes $|\text{Tree}_{student}|$ steps to delete the student AST and $|\text{Tree}_{canonical}|$ steps to build the canonical AST.

when "*total rainfall* `is greater than` *absorption limit*" (c.f. Figure 4.3 and Algorithm 3 in Section 4.1).

Table 6.10: Descriptive statistics of the scores of the runoff computational models

	r1_absorption	r1_runoff	r2_absorption	r2_runoff	r3_absorption	r3_runoff
mean	0.91	0.99	0.86	0.95	0.88	0.72
std	0.29	0.11	0.35	0.22	0.32	0.45
min	0	0	0	0	0	0
25%	1	1	1	1	1	0
50%	1	1	1	1	1	1
75%	1	1	1	1	1	1
max	1	1	1	1	1	1

An interquartile range (IQR) rule was applied to remove the outliers ($n = 11$), i.e., students who performed a large number of edits or tests to their computational models. Figure 6.4 shows the distribution of the behavior and performance measurements in the computational model-building activity as well as their pair-wise linear regressions. Table 6.11 lists the correlation coefficients between these measurements. As shown in Figure 6.4, the variables are not all normally distributed even after the removal of the outliers. Therefore, Spearman's ρ was used as the correlation coefficients in Table 6.11.

As shown in Figure 6.4 and Table 6.11, students' learning behaviors are correlated. For example, the number of edits and the number of tests to their computational modes had a Spearman's ρ of 0.47. On the other hand, only the number of tests to the computational model correlated with the computational model score. These results suggest that the students achieved higher modeling performance if they tested more frequently. But testing more frequently is also correlated strongly with the computational model edits. In other words, the more variations of the model, they also ran more tests, and those who ran more tests got better models. As most students were able to build a correct runoff model, the *computational_model_score* variable was skewed, which resulted in very small correlation coefficients with other behavior measurements.

Table 6.11: Spearman's ρ's of students' behavior and performance measurements in the computational model-building activity (statistically significant correlation coefficients ($p < 0.05$) marked in bold font)

	comp_edits	comp_tests	edit_between_tests
comp_edits	-		
comp_tests	**0.47**	-	
edit_between_tests	**0.29**	-0.08	-
comp_score	0.00	**0.42**	0.00

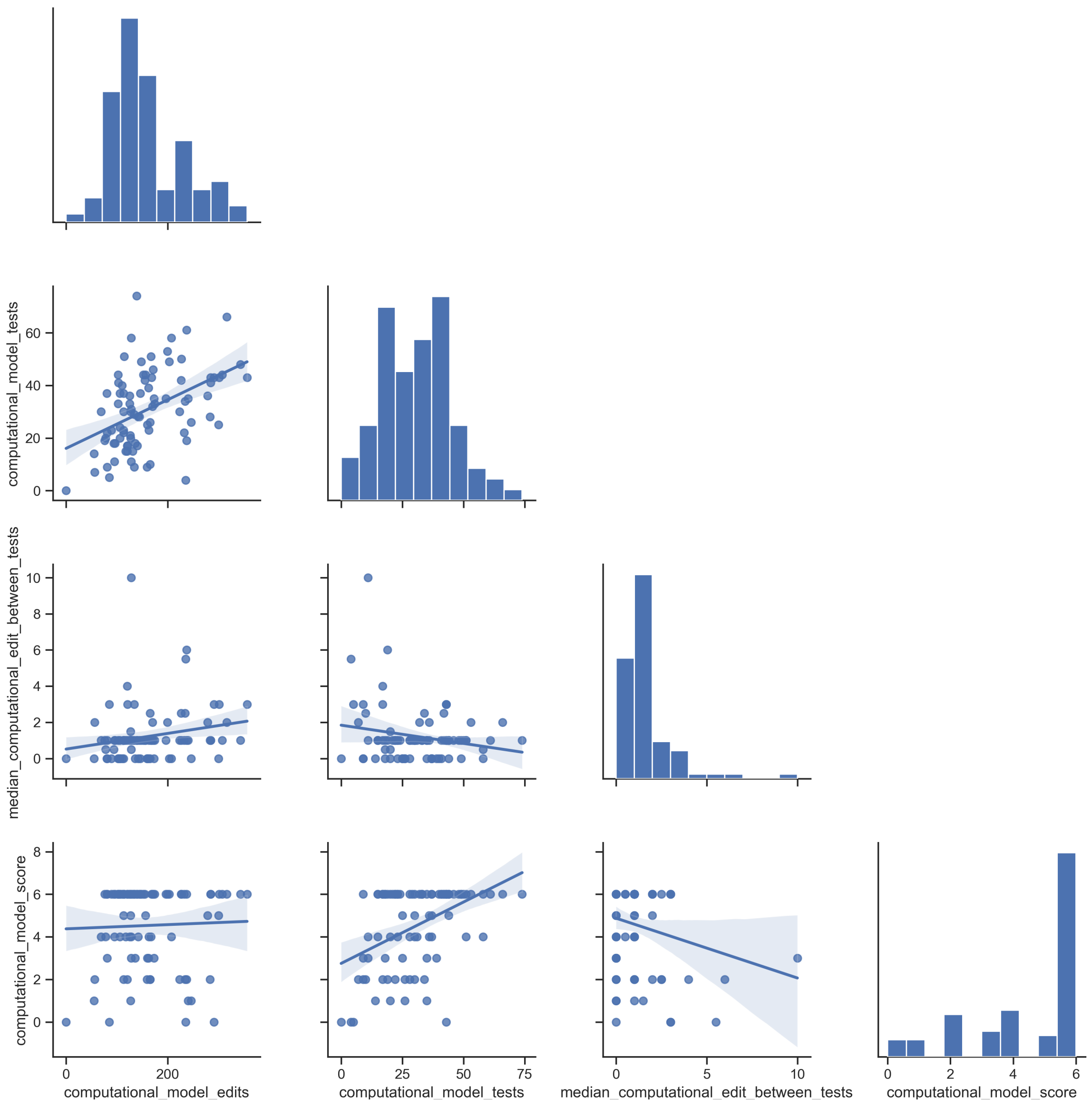


Figure 6.4: Students' behaviors and performances in the computational model-building activity

Finally, to evaluate the effect of the *bottom-out* hints given to the students before they proceeded with the engineering design activity, we used the non-parametric Mann-Whitney U-test on the runoff model scores before and after showing the solution to the students. The test statistics were Mann-Whitney $U = 4284$, $p = 0.11$. The small difference between the before-and-after model scores indicates that most of the students were able to generate runoff computational models that were close to the correct implementations on their own.

6.4 Engineering Design Performances

Table 6.12 summarizes the descriptive statistics of the variables measured in the engineering design activity. During the engineering design activity, the students performed an average of 29.4 tests (*SD* = 22.2) on their schoolyard designs. The average total standardized Euclidean distance was 18.6 (*SD* = 19.0). The average number of unique designs that satisfied the criteria for cost and accessibility was 6.3 (*SD* = 4.2). Eighty-nine students created and tested at least 1 satisfying design, and the average amount of runoff for the satisfying design solutions, with 2 inches of rainfall, was 1.23 inches (*SD* = 0.94). The global minimal runoff of all satisfying designs was 0.96 inches, and 29 students arrived at this optimal solution. These results show that the students were also able to create feasible design solutions.

Table 6.12: Descriptive statistics of the variables in the engineering design activity

	engineering test	unique satisfying design	lowest runoff
mean	29.38	6.31	1.23
std	22.19	4.25	0.94
min	0	0	0.9625
25%	13.5	3	0.9625
50%	24	6	1.06
75%	37	9	1.14
max	120	17	2.0

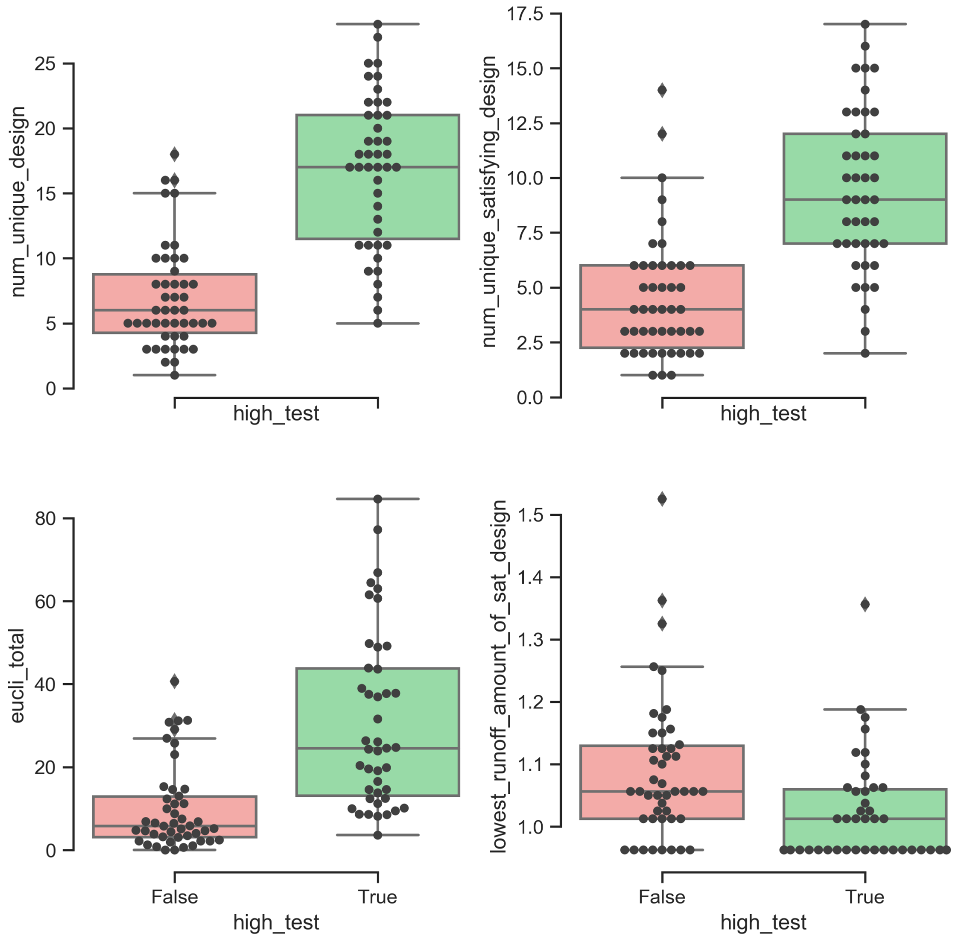


Figure 6.5: Box plots (with marked data points) of students' behaviors and performances in the engineering design activity (high-testing group $n = 43$ and low-testing group $n = 46$)

For the succeeding analyses in this section, data was removed for the 10 students who failed to generate a satisfying schoolyard design throughout this activity. We divided the remaining students into two two *ad hoc* groups according to the number of tests of their schoolyard designs using a median split (high-test (n = 43) and low-test (n = 46)). Figure 6.5 shows the box plot visualizations of four variables of (1) the number of unique designs tested, (2) the number of unique satisfying designs, (3) the total Euclidean distance between the tested designs, and (4) the lowest amount of runoff of satisfying designs[3]. Among these four variables, #1 and #3 are behavioral measurements; and #2 and #4 are performance measurements. The high- and low-testing groups are colored in green and red hues in Figure 6.5.

Hypothesis test results based on the measured differences among the high-test and low-testing groups are reported in Table 6.13. The non-parametric effect sizes are calculated as $\frac{\text{Mann-Whitney } U}{n_1 n_2}$, where $n_1 = 43$, $n_2 = 46$ after the adjustment of removing the outliers.

Table 6.13: Mann-Whitney U test results of learning behaviors in the engineering design activity

measurement	p-value	effect size
num_unique_design	< 0.001	0.09
euclid_total	< 0.001	0.13
num_unique_satisfying_design	< 0.001	0.15
lowest_runoff_amount_of_sat_design	< 0.001	0.30

As measurements of students' learning behaviors, both the number of unique tested designs (*num_unique_design*) and the total Euclidean distance between the tested designs (*euclid_total*) are significantly different between the high- and low-testing students with small effect sizes. These results suggest that the effort in making and testing more schoolyard designs positively influenced the students' learning performances in the engineering design activity.

[3]During the current classroom implementation, the teachers preferred framing the cost and accessibility criteria as *pass / fail* instead of a optimization as *minimizing cost / maximizing accessibility*. Therefore, the engineering design score as introduced in Section 4.6 consists only of the runoff criterion.

Table 6.14: Spearman's ρ's between engineering performance and engineering behaviors

		low-testing group		high-testing group	
behavior variable	performance variable	Spearman's ρ	p-value	Spearman's ρ	p-value
num_test	num_unique_satisfy	**0.41**	0.004	**0.41**	0.007
num_unique_design	num_unique_satisfy	**0.81**	<0.0001	**0.66**	<0.001
euclid_total	num_unique_satisfy	**0.52**	0.0002	0.27	0.085
num_test	lowest_runoff_amount	-0.16	0.281	-0.04	0.814
num_unique_design	lowest_runoff_amount	**-0.31**	0.037	-0.19	0.218
euclid_total	lowest_runoff_amount	-0.29	0.053	-0.24	0.124

To further investigate the relationship between students' learning behaviors and performances, we calculated the correlations between the behavior measurements (*num test*, *num unique design*, and *euclid total*) and the performance measurements (*num unique satisfying design* and *lowest runoff amount of sat design*). Figure 6.6 visualizes the paired relationships using the high- low-testing group split.

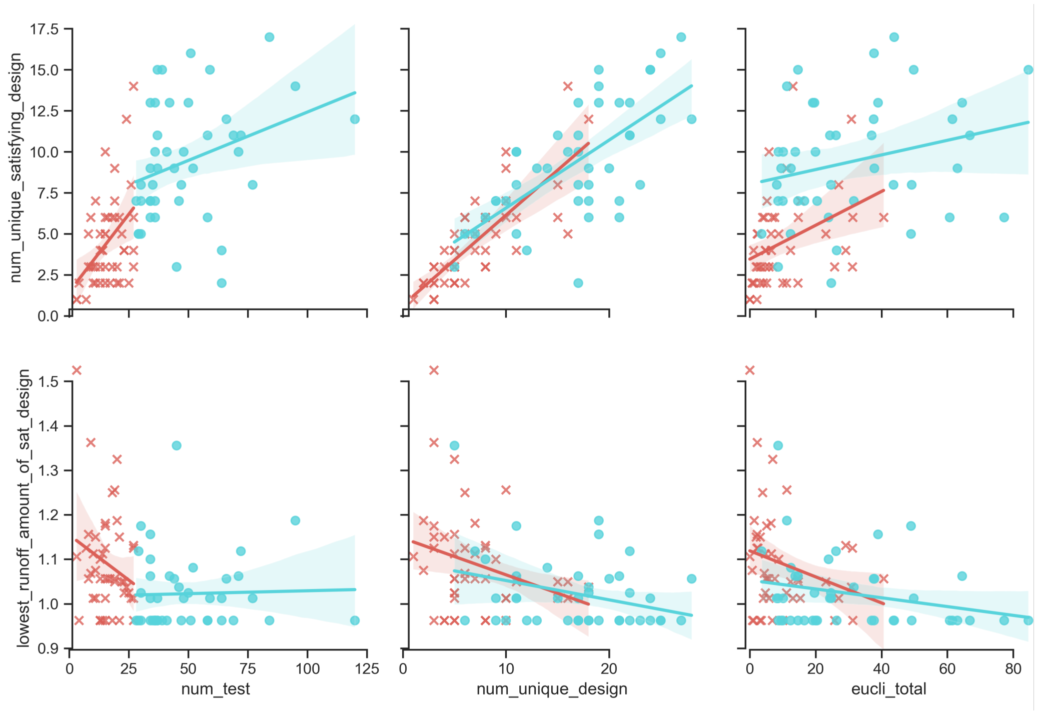


Figure 6.6: Paired regression plots of students' behaviors (x-axis) and performances (y-axis) in the engineering design activity (red: low-testing students; blue: high-testing students)

As shown in Table 6.14, for the low-testing group, four out of six pair-wise correlations are significant, and all of the pairwise correlations are high for the low-testing group as compared to the high-testing group. Interestingly, only two of the correlations were significant for the high-test group. The blue dots and regression lines (of the high-testing group) in Figure 6.6 further indicate that for the high-testing group, the behavior variables were related to the number of unique satisfying designs generated (these slopes were greater than for the low-test group), but the behavior variables did not seem to significantly affect the generation of the best runoff solutions (bottom row).

These results suggest that students' effort in the engineering design activity contribute to the learning performances *in general* (measured by the quality of the schoolyard designs), which further strengthen the results shown in Figure 6.5, i.e., (1) students who tested more are more likely to discover more satisfying designs, and (2) students who tested more tended to derive satisfying designs that reduced the runoff more. On the other hand, an excessive amount of designs and tests does not further contribute to the learning performances. As the regression coefficients of the high-testing group flattened out quickly and became not significant. These findings call for further investigations in the learning strategies applied by the students.

6.5 Path Analysis

Using the hypothesized structure as shown in Figure 5.9 in Section 5.3, we created a path diagram to further study the relations between the measured variables using the IBM® SPSS® Amos 26 software. We modeled a total of 47 direct effects from the 15 variables in the path diagram. As a pre-analysis suggested by Schreiber et al. (2006), we evaluated the assumptions of multivariate normality and then removed four outliers from subsequent analyses, leaving a sample size of 95 for the Path Analysis. Bootstrap samples ($N = 1000$) were generated to estimate the standard errors and calculate the confidence intervals at the 95% level. The standard errors and their critical ratios were later used to evaluate the statistical significance of the modeled causal effects while reducing the variance in the observed variables.

We also calculated the model-fitting statistics of the path model as compared to the saturated model that includes pairwise associations among all variables (Schreiber et al., 2006): $\chi^2 = 40.89$ (DF=54, p-value=0.91); the goodness of fit (GFI) was 0.95 ($\geq$ 0.95 threshold); the comparative fit index (CFI) was 0.99 (> 0.9 threshold); and the root mean square error of approximation (RMSEA) was 0.01 (< 0.06 threshold). These statistics indicate that the path model fitted the measurements well. All of the hypothesized paths in Figure 5.9 were confirmed as direct or indirect effects. Figure 6.7 shows the statistically significant causal paths that are non-trivial (having a direct standardized effect of $|\beta| > 0.1$).

The total standardized effect of an exogenous variable on an endogenous variable is calculated as the sum of the direct effect and the multiplications of all direct effects on a path ($\Sigma\beta$). Table 6.15 reports the total standardized effects of the variables. Each cell can be interpreted as the total effect of the *column* variable on the *row* variable, therefore, the table also presents the full structure of the path diagram in addition to the direct effects visualized in Figure 6.7. We check the direct and indirect effects on the categories of variables in Sections 6.5.1, 6.5.2, and 6.5.3 below.

Table 6.15: Standardized total effects of the learning behaviors and performances in the WRC

index & variable	1 pre sci	2 pre eng	3 pre ct	4	5	6	7	8	9	10	11	12
4 formative	.08	**.20**	**.30**									
5 comp edit	.02	-.16	.04	**.19**		**-.08**	**.26**					
6 comp test	.03	**.06**	**.28**	**.31**								
7 edit btw test	.00	.00	-.05	.01		**-.30**						
8 comp model score	-.03	**.12**	**.28**	**.54**	-.07	.14	**-.24**					
9 engineering test	.01	**.22**	.07	.06		.18						
10 eng euclid	.02	.14	.08	**.20**		.08			**.47**			
11 num satisfy	.00	**.16**	-.06	.12	-.01	**.14**	**-.04**	.18	**.64**	**.25**		
12 lowest runoff	.01	**-.11**	-.05	**-.15**	.02	**-.11**	**.08**	**-.32**	**-.34**	-.11	**-.35**	
13 post sci	.02	-.02	-.04	-.17	-.01	**.04**	-.04	.18	**.08**	.03	**.08**	**-.23**
14 post eng	.00	**.54**	.01	.09	-.01	.02	-.03	.11	.01	**-.20**	**.20**	-.01
15 post ct	.00	**.24**	**.55**	.10	-.03	.02	-.04	**.14**	-.01	.00	-.01	.02

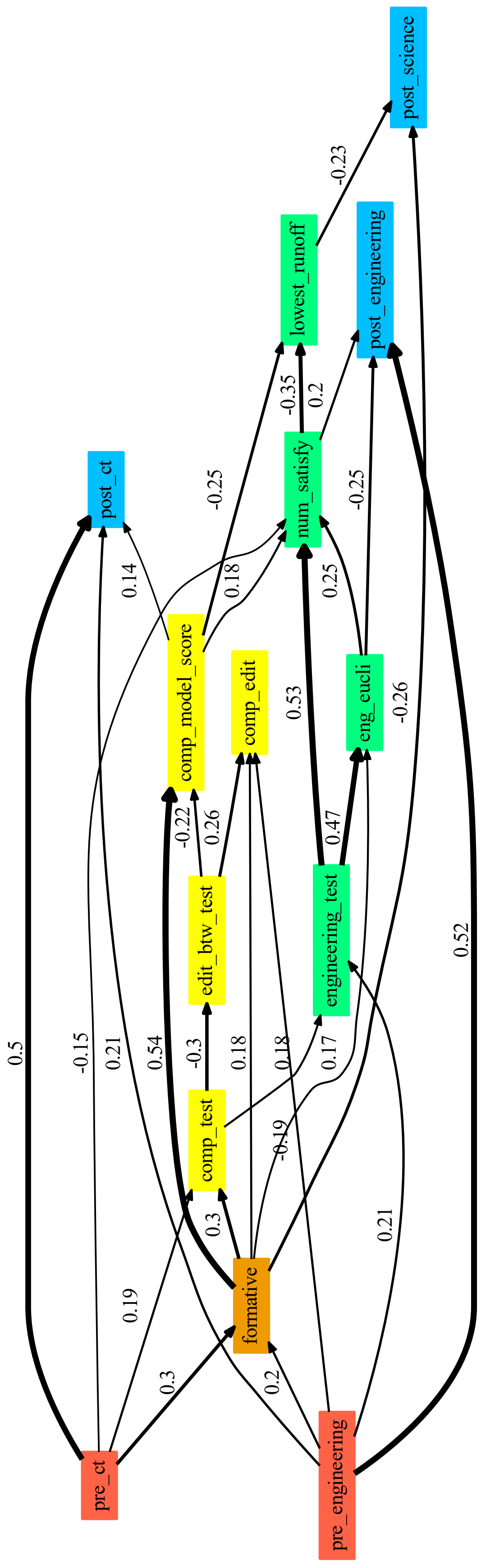


Figure 6.7: Discovered causal paths with statistically significant direct effects

6.5.1 Effects on Computational modeling

The students' learning behaviors and performance in the computational modeling activity (yellow boxes in Figure 6.8) were directly affected by variables in the same category. The associations within the behavior and performance variables in the same category have been discussed in terms of correlation analysis in Section 6.3. Figure 6.8 shows the subsection of the discovered causal paths up to the computational model-building variables.

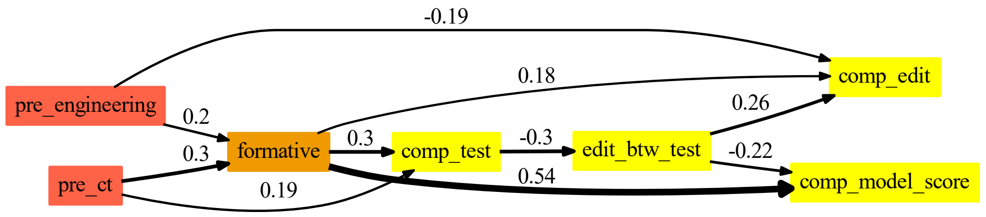


Figure 6.8: Effects on the computational model-building measures

The formative assessment score (*formative*) had significant effects on three of four of these variables directly. The direct and total effects from *formative* to *comp_test*, *edit_btw_test*, *comp_edit*, and *comp_model_score* were 0.3, 0.1, 0.18, and 0.54, respectively. These results, especially with the high η^2 of 0.29 between *formative* and *comp_model_score*, show that the formative assessment tasks closely mirror the curricular activities in the WRC. Furthermore, the formative assessments are a strong indicator of students' learning outcomes, and contribute to nearly 30% of the variance in the computational model score can be explained by the integrated formative assessment score.

In addition, the CT pre-test score also directly or indirectly affected the *comp_model_score* and *comp_edits* (via *formative*, *comp_test*, and *edit_btw_tests*) with $\Sigma\beta$'s of 0.28 for both (indirect effects are not shown in Figure 6.7 but can be read from Table 6.15).

As one of the main learning outcomes, the students' *comp_model_score* was affected by the students' engineering pretest score *pre_engineering*, (indirectly, $\Sigma\beta = 0.12$). It was also significantly affected by the median number of model edits between tests (*edit_btw_test*, indirectly, $\Sigma\beta = -0.24$), indicating students who edited their model in *smaller* chunks between tests performed better in the computational model-building task. Similar results of smaller edit chunks being associated with better modeling performance have also been reported by Basu et al. (2017).

6.5.2 Effects on Engineering design

As of the engineering design activity (green boxes in Figure 6.9), the number of unique satisfying designs (*num_satisfy*) and the lowest amount of runoff of satisfying designs (*lowest_runoff*) were the two variables evaluating the quality of students' designs. These two variables were significantly affected by numerous exogenous variables.

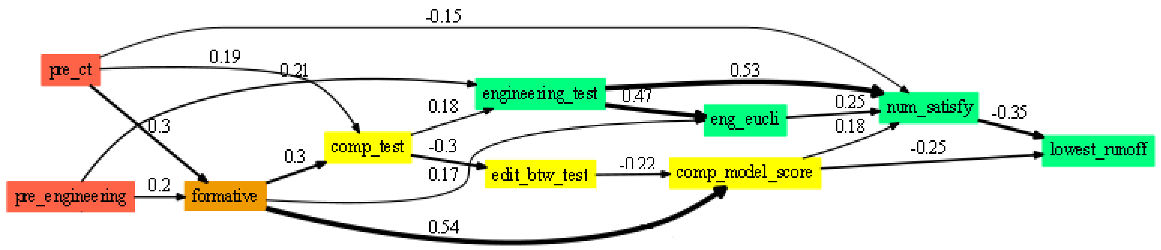


Figure 6.9: Effects on the engineering design measures

For *num_satisfy*, the strongest direct effects came from the number of tests on the designs (*engineering_test*, β=0.53) and the total standardized Euclidean distance between the tested designs (*eng_euclid*, β=0.25). The *lowest_runoff* was most strongly affected by *num_satisfy* (β=-0.35), *comp_model_score* (β=-0.25), and *formative* (mainly indirectly, $\Sigma\beta = -0.15$). In addition, *engineering_test* had a moderate total effect on *lowest_runoff* ($\Sigma\beta$'s=-0.32), and this effect mainly came via *num_satisfy*.

These results align with our findings from a previous study with fifth-grade students in another school. Students who explored a larger portion of the problem space systematically were more likely to generate better engineering design solutions (Zhang et al., 2019). It also matched the **scientific discovery as dual search** theory (Klahr and Dunbar, 1988), in that successful learners connect the **hypothesis space** and the **experiment space** by making inferences with data drawn from their investigations. More importantly, these results also suggest a strong connection between computational modeling (*comp_model_score*) and engineering design (*lowest_runoff*) with a total standardized effect of -0.32 ($\beta = -0.25$, the total indirect effect is -0.07). The negative value indicates that students making better computational models on their own generated better design solutions, even though all students were shown the correct implementation of the computational model before the engineering design activity. It also indicates the benefits of having students develop their own computational model to use for designing and testing, relative to providing students with a model that has been developed by experts.

6.5.3 Effects on Post-test scores

The post-test scores (blue boxes in Figure 6.10) were affected by all other categories of variables in the WRC learning activities and most significantly by the pre-test scores as exogenous variables (c.f. the correlation analysis reported in Section 6.1.1).

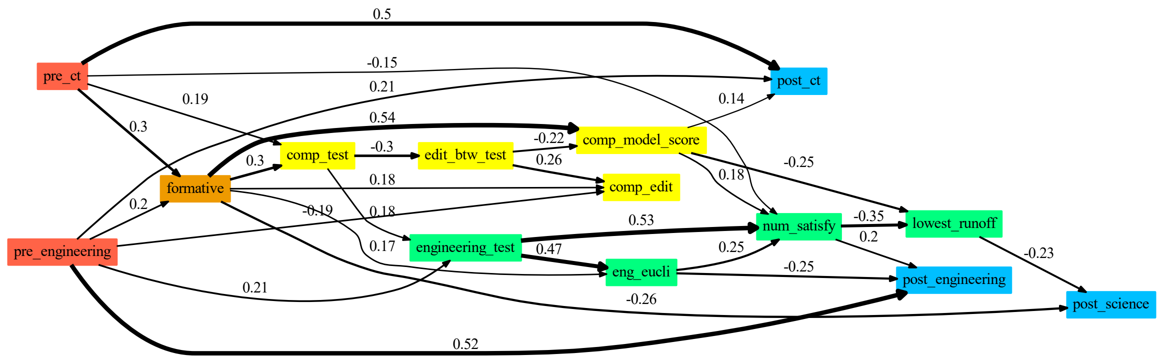


Figure 6.10: Effects on all measures

The **science** post-test scores (*post_sci*) were significantly influenced by *lowest_runoff* (directly, $\beta = -0.23$), *num_satisfy* (indirectly, $\Sigma\beta = 0.08$), *engineering_test* (indirectly, $\Sigma\beta = 0.08$), *comp_ model_score* (indirectly, $\Sigma\beta$=0.04), and *formative* (directly, $\beta = -0.26$). The effect from the engineering design activity (*lowest_runoff*) on science learning and understanding (*post_sci*) again demonstrated students' synergistic learning and the effectiveness of the design principle of integrating engineering learning with science that has been called out in the NGSS framework (NAE and NASEM, 2019).

The **engineering** post-test scores were mostly affected by *pre_eng* (directly, β=0.52), *eng_euclid* (directly, β=-0.25), and *num_satisfy* (directly, β's=0.20). The effects of both the learning behaviors (*eng_euclid*) and the performance (*num_satisfy*) indicate that students' success in solving the engineering design problem by searching for the optimal combinations of surface materials on the schoolyard contributed higher learning outcomes. This finding again demonstrated the benefit of the WRC as a curricular unit for young novice learners.

As for the **CT** post-test score, it was only significantly affected by the related pre-test scores. The variable *comp_model_score* had a relatively large total effect of 0.14 on *post_ct* yet the effect was not statistically significant. Nevertheless, this effect shows a positive trend that students' performance in the computational model-building activity contributed to their CT learning.

These overall positive results suggest that the students' success with the engineering design activities can be linked to their science and engineering proficiency, providing evidence for the benefit of integrating engineering with science learning (NAE and NASEM, 2019). In addition, the effect of engineering activities on the summative assessments suggests that the design goals of the WRC curriculum were achieved.

6.6 Summary of Chapter

This chapter investigates three research questions of this dissertation research. We report the students' learning outcomes in the WRC curriculum, including (1) the domain and CT pre-post test gains and their performance in the integrated formative assessments, (2) the behaviors and performances in the computational modeling activity, and (3) the behaviors and performances in the engineering design activity. Overall, these learning outcomes illustrated the successful implementation of the WRC and its instructional benefits across science, engineering, and CT.

The results from the analysis of students' summative assessment scores suggest that (1) the sixth-grade students achieved significant and large learning gains (Cohen's $d = 1.02$) overall, (2) both teachers effectively delivered the WRC curriculum ($\eta_p^2 = 0.002$), (3) the students' prior knowledge had an effect ($\eta_p^2 = 0.233$) on the learning outcome, and (4) the significant correlations between the science, engineering, and learning gains in CT suggest synergy of learning. In addition, the effect of engineering activities on the summative assessments suggests that the design goals of the WRC curriculum were achieved, and students' high learning gains (Cohen's d=1.02) demonstrate the benefits and promise of integrating instruction across engineering and science.

Second, with our investigation of *RQ 2* using the Path Analysis, the results from both the computational model-building and the engineering design activities suggest that students' performance in WRC (as measured by the model scores, number of satisfying designs, and the lowest amount of runoff) is strongly associated with their respective learning behaviors. On the other hand, we observed a negative influence on the learning performance when the students committed an excessive amount of edits or tests to both the computational model and the schoolyard design.

Third, we identified the connections between computational modeling, engineering design, and the learning outcomes as *effects* on the causal paths. Such connections might not be discovered by only examining the associations between the variables using model-less correlation methods (Pearl and Mackenzie, 2018). For example, the correlation coefficient (Spearman's ρ) between *comp_model_score* and *lowest_runoff* was -0.11 ($p = 0.28$). This suggests that Path Analysis is an effective technique to study the relationship between related variables, such as the measures derived from the WRC.

Previous analyses using a subset of the results in this chapter have been reported in Zhang et al. (2020).

Chapter 7

Students' Strategy Use in the WRC Curriculum

When students work on complex problems, they develop and apply a variety of strategies to support their learning and problem-solving tasks. For example, students often reuse solutions derived for previous problems and may apply trial-and-error approaches to advance when they encounter difficult situations. In this context, strategies represent students' *conscious and controllable sequences of actions to facilitate and enhance task performance* (Zhang et al., nd).

The use of strategies is latent, thus, they cannot be observed or measured directly. However, the use of strategies manifests students' declarative, procedural, and conditional knowledge (Schraw et al., 2006). As stated in Chapter 3 of this dissertation, we hypothesize that the use of strategies is reflected in students' observable behaviors that can be derived from their logged activity data.

Previous work (e.g., Segedy et al. (2015b,a); Zhang et al. (2017) suggested that the aggregated description of behaviors (e.g., the time spent on certain learning activities) can mask the exact manifestation of strategies because of the loss of contextual information in aggregated actions (Kinnebrew et al., 2017). Therefore, to retain context information when analyzing students' learning behaviors, and to make it easier to identify strategies, we have proposed the use of mining algorithms that support finer-grained analyses of student behaviors such as Markov Chain (MC) models as proposed in Section 5.4.

We model, track, and analyze students' development and use of learning strategies during the computational model-building and engineering design processes in Sections 7.1 and 7.2 to answer *RQ 4*, i.e., *How do students utilize strategies to facilitate their learning processes?*

7.1 Analyses of Students' Computational Model-building Processes

In this section, we (1) examine students' actions in the computational model-building activity, (2) present and discuss MC models, and (3) link students' learning behaviors to their learning performances with the MC models.

7.1.1 Markov Chain Modeling of Students' Computational Modeling Activities

As presented in Section 4.5, the actions in the computational model-building activity include `AddBlock`, `ConnectBlock`, `DisconnectBlock`, `RemoveBlock`, `FillField`, and `StartSimulation`. A total of 23,805 actions were logged during the computational model-building processes. We removed the outliers whose time interval between their succeeding actions is either too short (less than 10 milliseconds, indicating a batch operation such as undo or redo in the NetsBlox environment) or too long (greater than 40 minutes, indicating the end of the day). We then used the 95th percentile of the remaining 23,319 actions (80 seconds) to determine the threshold of an idling action. As a result, an implicit `Idle` action will be added to the sequences of each student's actions if they did not perform any activity for at least 80 seconds when they worked on the system. Table 7.1 presents the descriptive statistics of the distribution of the time intervals.

Table 7.1: Descriptive statistics of the time intervals during the computational model-building activity (total number of time intervals = 23,319)

	Interval
mean	0:00:21.33
std	0:01:20.20
min	0:00:00.01
25%	0:00:02.20
50%	0:00:05.03
75%	0:00:13.11
Idle threshold (95%)	0:01:19.99
Disengaged threshold (99%)	0:07:32.48
max	0:39:06.68

Notably, the meaning of `Idle` actions is context-dependent. Students could be reflecting on their problem-solving process, interpreting the simulation result, or simply becoming disengaged. We also introduced a `Disengaged` action to separate a valid idle action and the end of class if the time interval is longer than the 99th percentile of action gaps. The `Disengaged` action can alternatively be interpreted as the end of the class or the start of a new class day (SoD). Table 7.2 presents the number of instances of the actions and their proportions. At the highest level, building (`ConnectBlock`) and testing (`StartSimulation`) the computational model were the most frequent types of actions.

Table 7.2: Count and proportion of actions in the computational model-building activity

Action	Number of instances	Proportion
AddBlock	1233	0.050
ConnectBlock	8876	0.361
DisconnectBlock	1932	0.079
RemoveBlock	1629	0.067
FillField	2453	0.100
StartSimulation	6970	0.284
Idle	1144	0.047
Disengaged	118	0.005

These results in Table 7.2 show that the students performed clustered computational model edits. The short intervals between actions overall (75th percentile = 13.11 seconds) also indicated that the students did not spend much time reflecting between model-building actions. Therefore, it is important to examine students' action sequences in finer granularity.

We derived a first-order Markov Chain model of all participants (n = 99) by (1) using the proportions of actions listed in Table 7.2 as *prior probabilities* of each state in the MC model and (2) the number of instances of any consecutive pairs of actions as the conditional probability, i.e.,

$$\frac{Count(Action_t = A \wedge Action_{t+1} = B)}{Count(Action_t = A)}, \ \forall t \in \{T\}, \forall A, B \in \{ModelBuildingActions\}$$

Figure 7.1 shows the MC model for the computational model-building activities (the GraphViz software (Gansner and North, 2000) was used to generate this figure). Each state in the model represents one of the actions of `AddBlock`, `ConnectBlock`, `Disconnect Block`, `RemoveBlock`, `FillField`, `StartSimulation`, `Idle`, and `Disengaged`. The size of the states is proportional to the number of occurrences of the particular action (c.f. Table 7.2). An arrow between two states represent a transition, and the number on the arrow represents the conditional probability associated with that transition, i.e., the first-order Markov transition probability. Arrows with transition probabilities less than 0.09 are not visualized in Figure 7.1 for clarity. The topology of the states is also determined by the transition probabilities. Table 7.3 shows the transition matrix of the Markov Chain model for all students ($n = 99$). The transitions with a probability > 0.2 are displayed in the bold format in Table 7.3.

Table 7.3: Transition probabilities of students' computational model-building processes

	AddBlk	ConnectBlk	DisconnectBlk	FillField	RemoveBlk	StartSim	Idle	Disengaged
AddBlk	**0.208**	**0.537**	0.039	0.009	0.102	0.081	0.021	0.002
ConnectBlk	0.035	**0.567**	0.091	0.084	0.055	0.132	0.031	0.003
DisconnectBlk	0.021	**0.488**	**0.234**	0.017	0.115	0.109	0.013	0.000
FillField	0.024	**0.206**	0.025	0.137	0.020	**0.546**	0.033	0.002
RemoveBlk	0.112	**0.401**	0.044	0.033	**0.242**	0.104	0.053	0.004
StartSim	0.021	0.115	0.054	0.149	0.039	**0.508**	0.094	0.011
Idle	0.101	**0.221**	0.089	0.190	0.065	**0.335**	n/a	n/a
Disengaged	0.169	**0.220**	0.085	0.136	0.025	**0.364**	n/a	n/a

As shown in Table 7.3, `ConnectBlock` is a very frequent action in the computational model-building activity and it made up 36.1% of the total amount of actions. In Figure 7.1, `ConnectBlock` bears the structure of a *sink* state on the Markov Chain model, as it was a very likely subsequent action for all actions, especially model edit actions such as `AddBlock`, `DisconnectBlock`, and `RemoveBlock`.

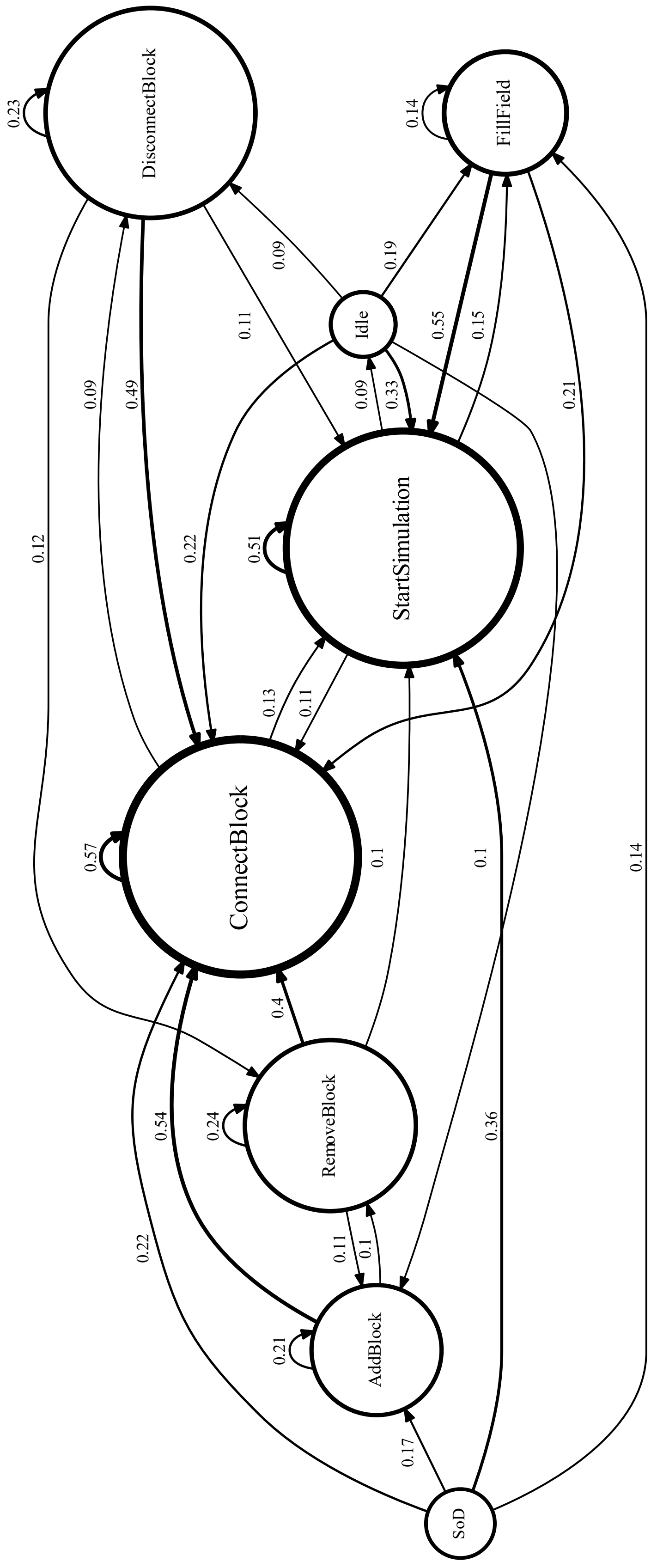


Figure 7.1: Discrete-time Markov chain model of students' computational model-building processes

`StartSimulation` was another frequent action that accounted for 28.4% of all the actions. This action provides key information for the learner to estimate the correctness of their models by examining the values of the output variables (the amounts of absorption and runoff, c.f. Figure 4.4). Therefore, it is a central step to construct correct computational models especially in the *debugging* steps. As shown in Figure 7.1, actions frequently preceded `StartSimulation` included `FillField`, `Idle`, `Disengaged`, and the action itself (implying a self-loop).

The `FillField` action was the third most frequent action during the computational modeling activity. In the modeling process, it mainly appeared in two contexts: (1) as part of the initial process building the model and (2) as part of tweaking parameters to test the model. The transitions between `FillField`, `ConnectBlock`, and `StartSimulation` correspond to these two contexts.

In summary, the results in this section indicate that MC modeling is a concise and indicative representation of students' learning behaviors at an aggregate level. In the next step, we examine the links between the individual students' learning behaviors to their learning performance.

7.1.2 Linking Computational Modeling Behaviors to Learning Performance

For research question 4 of this dissertation research (i.e., *How do students utilize strategies to facilitate their learning processes?*), we hypothesized that the differences in certain action sequences can be indicative of the students' learning performance. Using the same method to fit a Markov Chain model, we derived MC models for each individual student. A Laplace Smoothing (Bishop, 2007) was applied to avoid likelihoods of zero in the empirical data if a transition did not occur for certain students (for example, some students never idled).

Because Markov Chain is a generative model, we can estimate the likelihoods of sequences of actions (e.g., `StartSimulation` → `Idle` → `RemoveBlock`) generated by each individual Markov Chain model. We can then use the generated likelihoods to predict learning performances and associate the action sequences with the learning performances using regression methods. As the first step, we calculated the likelihoods of pairs and triplets of actions generated by individual MC models for all 99 students[1]. We then applied multivariate regression analyses to check if these likelihoods predict the students' learning gains in the summative assessments. Table 7.4 lists the statistically significant predictors (likelihoods of action pairs) of the normalized learning gains. The regression coefficients are standardized (denoted as β) and ranked in descending order. Therefore, both the sign and the magnitude of the coefficients can be used to discuss the link between certain learning behaviors and the learning outcome. In addition, we calculated the lift measures of the patterns as the ratio between the expected probability of an action relative to the probability of the action pair (Merceron and Yacef, 2008) as described in Section 5.4. Sequences #1, #5, #6, #8 and #9 in Table 7.4 had either large lift measures greater than 1 or small lift measures close to 0, thus, they were considered *interesting*.

Table 7.4: Statistically significant predictors (likelihoods of pairs of actions) of the learning gains

	action sequence	β	*std err*	p-value	lift
1	DisconnectBlock→FillField	0.3952	0.142	0.008	0.17
2	Idle→ConnectBlock	0.3426	0.168	0.048	0.61
3	StartSimulation→DisconnectBlock	0.3326	0.147	0.029	0.68
4	ConnectBlock→Idle	0.3045	0.137	0.032	0.65
5	RemoveBlock→AddBlock	-0.2736	0.133	0.046	2.21
6	FillField→RemoveBlock	-0.2962	0.124	0.022	0.30
7	Idle→RemoveBlock	-0.3093	0.120	0.014	0.97
8	FillField→DisconnectBlock	-0.4093	0.118	0.001	0.31
9	StartSimulation→Disengaged	-0.6423	0.269	0.022	2.34

As shown in Table 7.4, the likelihood of some action sequences emerged as significant predictors of the learning gains. We observed four pairs of actions as positive predictors and five pairs of action as negative predictors and discuss them below.

[1]Longer action sequences can be seen as the combination of shorter sequences and thus provide limited added information. Therefore, we focus on the short sequences.

To begin with, one characteristic of the pairs of actions involves the role of the `Idle` action combined with the `ConnectBlock` action —the action with the highest number of instances among all students (c.f. Table 7.2). Both sequences #2 and #4 are positive predictors of the learning gains. As suggested by the time intervals listed in Table 7.1, the students spent overall, a very short amount of time between actions. As a result, less than 5% of the actions during the computational modeling activity had a gap longer than 80 seconds. It appeared that those who are more likely to pause and revisit the problem at hand (by idling for a short while) are more likely to achieve higher learning gains.

Second, as shown in Table 7.3, the probability of sequence #3 among all students (`Start Simulation` → `DisconnectBlock`) is 0.05. Because of the low probability, the edge is not visualized in Figure 7.1. Instead, a more likely sequence is through an intermediate idle state (i.e., `StartSimulation` → `Idle` → `DisconnectBlock`). The positive link between the likelihood of sequence #3 to the learning gains suggests that students who were able to quickly disconnect a block from the computational model after running a simulation (disconnecting a block may be considered to be part of a debugging action) understood the computational modeling task better and benefited more from it.

As for the negative predictors, sequences #5, #6, and #7 all include the `RemoveBlock` action, one of the least frequent actions during the computational modeling activity. On the one hand, the `RemoveBlock` action itself appeared to be an indicator of ineffective problem-solving process: if the block to be removed is incorrect or unnecessary, it must have been added or connected to the computational model as an ineffective model edit; otherwise, the removal of a correct block itself is not an effective action either and will have to be addressed separately. In addition, the more likely actions following `Idle` include `StartSimulation`, `ConnectBlock`, `FillField`, and `DisconnectBlock`, which all appeared as part of *debugging* or *tinkering* strategies as the student plans or reflects on the learning activity.

Similarly, in the computational modeling activity, the `FillField` actions mainly appear in two contexts: (1) as part of the initial process building the model and (2) as part of tweaking parameters to test the model. The high transition probabilities between the `FillField` action itself, `ConnectBlock`, and `StartSimulation` actions in the Markov Chain shown in Figure 7.1 confirm the two interpretations. On the other hand, the transition `FillField` → `DisconnectBlock` has a very low probability (0.025), and such an unnatural flow of actions suggests sub-optimal learning strategy use.

Finally, the negative link between the learning gains and the likelihood of the `Start Simulation` → `Disengaged` sequence suggests the undermining effect of being *disengaged* during the computational modeling activity. In this case, instead of using the simulation results to make an informed modification to the model (as part of the *debugging* or *tinkering* strategies), the student tested the model without doing anything until the end of the class.

On the other hand, this data-driven approach using generative likelihoods faces challenges as well. For example, the likelihood of the sequence `DisconnectBlock`→`Fill Field` is the strongest predictor of the learning gains. However, this pattern itself has a very low likelihood and does not belong to a natural flow of computational model edits. Its emergence as a positive predictor might be the result of over-fitting using relatively few numbers of instances even though the multiple-colinearity issue was controlled by the covariates of the predictors. This result indicates that the links between the learning behaviors and performances need to be corroborated by analyzing longer and more localized sequence patterns.

7.1.3 Difference in the MC Model

In this section, we investigate longer sequences of actions using the methods presented in Section 5.4. We divided the 99 students into two groups using a median cut of the total post-test score. Then for each action sequence to investigate, we generated two distributions of log-likelihoods using the MC models in the high-performance group (HG) and low-performance group (LG). Figure 7.2 shows the Markov Chain models for the HG and the LG students.

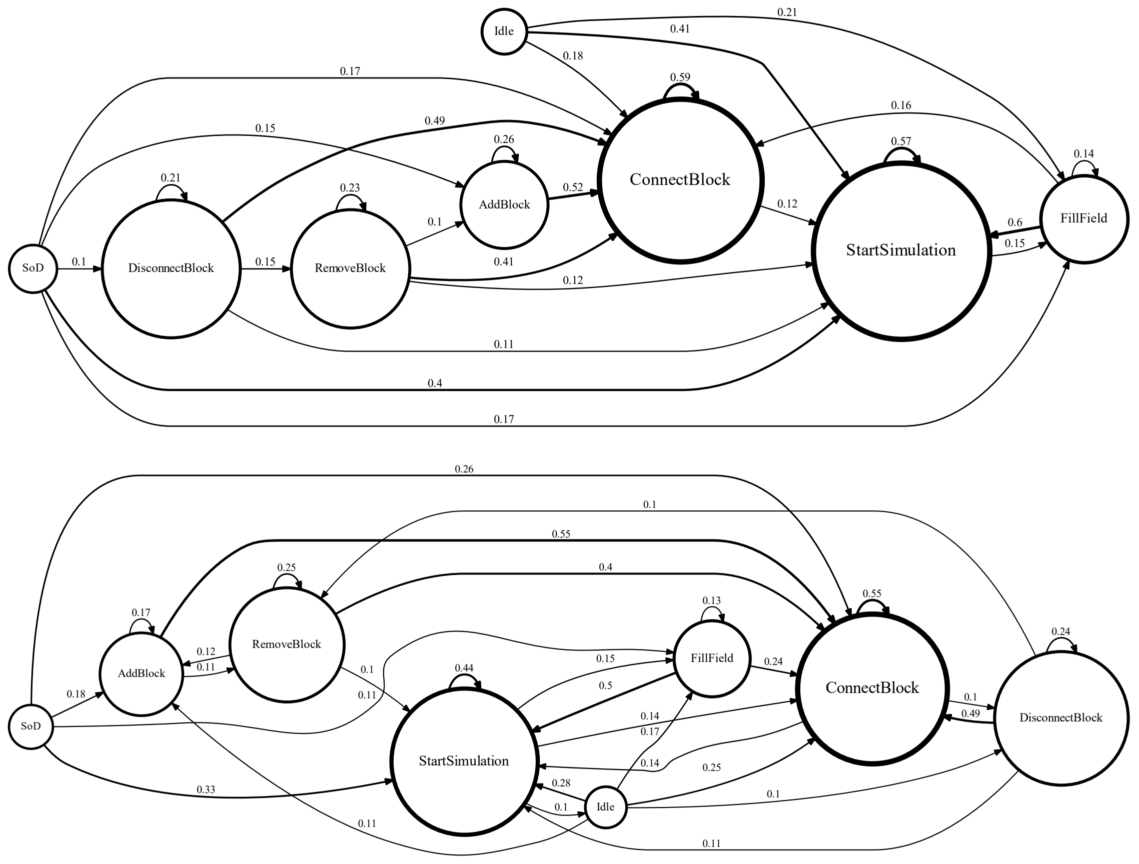


Figure 7.2: The Markov chain models of the computational model-building processes of the HG (upper image) and LG (lower image) students

Both the structure and the transition probabilities of the HG and LG models shown in Figure 7.2 differ greatly from each other. For example, the low-performing students (bottom) had a greater likelihood to transition between `AddBlock` and `RemoveBlock` and to transition between `ConnectBlock` and `DisconnectBlock`. The interweaving pattern is a typical behavior involving inefficient modeling approaches instead of applying *build-and-test* or *tinkering* strategies. In addition, the most frequent actions the high-performing students did at the start of the day is `StartSimulation`; whereas the low-performing students tended to directly make changes to the computational model with `AddBlock` and

`ConnectBlock`. These qualitative and quantitative differences among individual students' Markov Chain models serve as the foundation of investigating the links between students' learning behaviors and performances. In the rest of the section, we measure and analyze these differences using rigorous statistical test methods.

Table 7.5: Log-likelihood distributions (median and standard deviations of the log-likelihood) of high- and low-performing students and Mann-Whitney U-test results

	action sequence	HG (*Median, SD*)	LG (*Median, SD*)	diff	p-value	lift
1	DisconnectBlk→StartSimulation→RemoveBlk	-30.07 (13.92)	-7.91 (12.67)	-0.4	0.000	1.13
2	ConnectBlk→StartSimulation→RemoveBlk	-6.78 (11.83)	-5.85 (6.93)	-1.7	0.000	0.99
3	StartSimulation→RemoveBlk→AddBlk	-28.34 (16.04)	-6.62 (10.3)	-0.76	0.000	2.55
4	FillField→StartSimulation→StartSimulation	-3.79 (0.57)	-4.33 (5.7)	-9.8	0.001	1.18
5	StartSimulation→Idle→StartSimulation	-4.43 (3.39)	-5.13 (7.59)	-5.8	0.001	1.41
6	Idle→StartSimulation→StartSimulation	-4.8 (3.4)	-5.5 (5.42)	-3.3	0.002	1.37
7	ConnectBlk→RemoveBlk→AddBlk	-27.16 (12.88)	-6.12 (8.79)	-0.8	0.002	1.50
8	StartSimulation→StartSimulation→FillField	-4.03 (3.47)	-4.53 (3.42)	-6.9	0.004	0.89
9	RemoveBlk→AddBlk→ConnectBlk	-6.68 (12.73)	-5.49 (10.23)	-2.0	0.006	1.40
10	Idle→StartSimulation→FillField	-5.55 (5.08)	-6.39 (7.77)	-1.6	0.011	1.89
11	ConnectBlk→StartSimulation→DisconnectBlk	-6.79 (11.38)	-5.79 (7.44)	-2.5	0.013	1.24
12	FillField→ConnectBlk→StartSimulation	-6.45 (7.95)	-5.84 (4.57)	-1.9	0.032	0.60
13	FillField→StartSimulation→Idle	-4.67 (1.25)	-5.5 (7.9)	-4.6	0.037	3.34
14	Idle→StartSimulation→Idle	-5.78 (3.7)	-6.76 (7.69)	-1.5	0.042	3.89
15	ConnectBlk→ConnectBlk→ConnectBlk	-2.03 (0.7)	-2.27 (0.62)	-64.3	0.047	1.61

To determine if there is a significant difference in the distributions of the likelihoods, a non-parametric Mann-Whitney U-test is performed. To avoid Type I errors in the hypothesis test, we applied the False Discovery Rate adjustment (Glickman et al., 2014) by multiplying the original p-values by their rank until an $\alpha = 0.05$ cutoff was reached. Table 7.5 lists the hypothesis test results of the log-likelihoods of high post-test score students (HG) and low post-test score students (LG). Most of the lift measures of interestingness of the action patterns listed in Table 7.5 were larger than the threshold of 1.0. Additionally, we calculated the difference between the *i-frequencies* between HG and LG students (Kinnebrew et al., 2017)[2]. We summarize the differences in the rest of this section by grouping the 14 patterns by the implied strategy use of their most distinctive action component.

[2]the *i-frequency* is a measurement introduced in the Differential Sequence Mining algorithm that measures the average number of instances of an action sequence aggregated at the user level.

7.1.3.1 Block Reuse Strategy

Sequences #1, #2, #3, #7, and #9 all contained the `RemoveBlock` action. Among these sequences, the likelihoods of high-performing students to commit these actions were significantly **lower** than the low-performing students according to the median cut of the post-test scores. As presented in Section 4.5, both `DisconnectBlock` and `RemoveBlock` deactivate part of the computational model. Their difference is that while `RemoveBlock` makes the part completely out of the scope of its life cycle, `DisconnectBlock` preserves it so that the component may be reused later. Therefore, `RemoveBlock` is intrinsically associated with less proficient problem-solving strategies. This finding echoes the multi-regression analysis with pairs of actions that involved `RemoveBlock` (c.f. rows 5, 6, and 7 in Table 7.4. More specifically, high-performing students rarely performed sequences #1, #3, and #7, thus indicating their negative implications as a learning strategy.

To begin with, pattern #1 (DisconnectBlock → StartSimulation → RemoveBlock) is not an effective action sequence. As the first action in this sequence is to disconnect a block, a simulation could support the succeeding action to remove either the disconnected block or any other blocks. Second, pattern #3 shares the same ineffective transition of StartSimulation → RemoveBlock as pattern #1 and is followed by adding a block without acquiring information from `StartSimulation` either. Third, both pattern #7 and #9 appear to be inconsistent model edit actions without any testing even for the high-performing students. However, the likelihoods were significantly smaller. Finally, pattern #2 reflects the use of some *debugging* strategy and is a meaningful sequence of actions. These large differences in the median log-likelihoods among the high- and low-performers reflected the qualitative difference in students' strategic involvement with reusing blocks.

7.1.3.2 Reflection and Planning Strategy

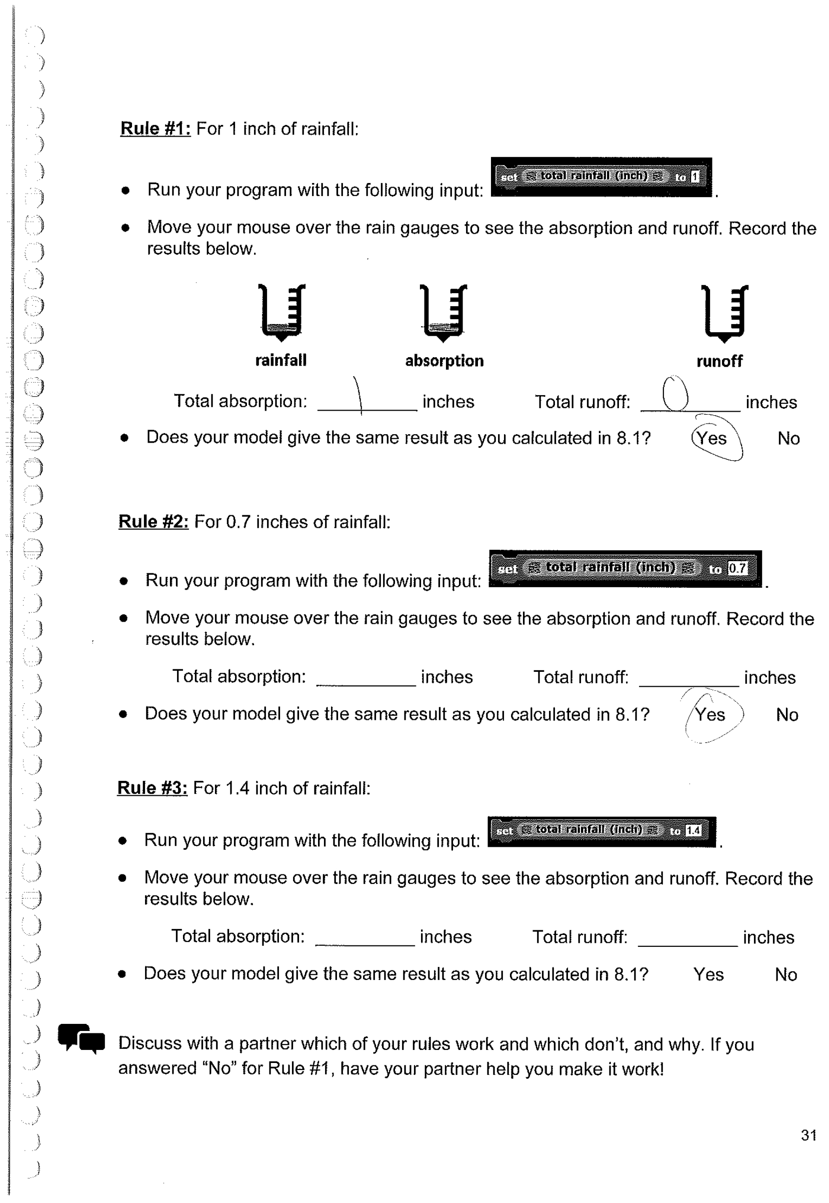
Patterns #5, #6, #10, #13, and #14 involved the implicit `Idle` action. The first observation that stood out in all these patterns is the **larger** likelihood for high-performing students. All of these patterns have a StartSimulation→Idle or Idle→StartSimulation component. Given the high lift measurements of patterns #13 and #14, as well as the short 80-second threshold of an `Idle` action in the computational modeling activity, we believe the high-performing students were likely to use cognitive and metacognitive strategies for their learning and modeling tasks (c.f., Section 2.5). By doing so, students could pause to (1) record their solutions and simulation output on their WRC notebook, (2) reflect on the simulation results, or (3) plan on the next steps of the modeling task instead of reflectively switch back to the model building actions.

In addition, pattern #15 (`ConnectBlock→ConnectBlock→ConnectBlock)` reflects a *depth-first* behavior of model construction. Typically, this type of behavior is common among students in the early phases of model building (Zhang et al., nd). On the other hand, HG students were less likely to perform consecutive `ConnectBlock` very much in contrast to the LG students (as indicated by the difference in i-frequencies).

On the other hand, students' idle and disengagement actions were estimated with a purely data-driven method. The understanding of students' modeling and problem-solving processes and strategy use can be improved by including multi-modal analyses with the screencast or eye-tracking data to measure the students' metacognitive strategy use more accurately.

7.1.3.3 Build then Test Strategy

Running simulations is a central process in developing and debugging computational models. All actions patterns in Table 7.5 except #7 and #9 include `StartSimulation`. Among these actions, #4, #8, #10, #12, and #13 involved the `FillField` actions, and the high-performing students had higher likelihoods in four out of five of these patterns. These results corroborate our previous finding in the regression analysis that testing the different parts of the computational model (c.f. Algorithm 3 in Section 4.1 and Figure 4.3 in Section 4.2) by tweaking values of the input variables could result in higher learning performance. Anecdotally, as a counter-example, Figure 7.3 shows a student's incomplete page where they only recorded the result from one testing condition of the computational model, and the same student had no *Idle* action during the three-day activity and had a negative learning gain in the post-test.



Rule #1: For 1 inch of rainfall:

- Run your program with the following input: `set total rainfall (inch) to 1`.
- Move your mouse over the rain gauges to see the absorption and runoff. Record the results below.

rainfall absorption runoff

Total absorption: 1 inches Total runoff: 0 inches

- Does your model give the same result as you calculated in 8.1? [✓ Yes] No

Rule #2: For 0.7 inches of rainfall:

- Run your program with the following input: `set total rainfall (inch) to 0.7`.
- Move your mouse over the rain gauges to see the absorption and runoff. Record the results below.

Total absorption: __________ inches Total runoff: __________ inches

- Does your model give the same result as you calculated in 8.1? [✓ Yes] No

Rule #3: For 1.4 inch of rainfall:

- Run your program with the following input: `set total rainfall (inch) to 1.4`.
- Move your mouse over the rain gauges to see the absorption and runoff. Record the results below.

Total absorption: __________ inches Total runoff: __________ inches

- Does your model give the same result as you calculated in 8.1? Yes No

Discuss with a partner which of your rules work and which don't, and why. If you answered "No" for Rule #1, have your partner help you make it work!

31

Figure 7.3: Incomplete student notebook activity

7.1.4 Summary

In this section, we discovered and interpreted three learning strategies involving block-reuse, reflection-and-planning, and build-then-test during the computational model building activity. In the next section, we apply the same method to detect and interpret the students' use of learning strategies in the engineering design activity.

7.2 Analyses of Students' Engineering Design Processes

The actions logged in the engineering design learning activity are `AddMaterial`, `RemoveMaterial`, `ResetDesign`, `RunSimulation`, and `ViewDesignHistory` (cf. Section 4.2). Table 7.6 lists the number of instances and the proportion of these actions. The types of actions are fewer than the computational modeling activity but the number of instances was much higher.

Table 7.6: Count and proportion of actions in the engineering design activity

Action	Number of instances	Proportion
AddMaterial	17613	0.491
RemoveMaterial	12327	0.344
RunSimulation	3460	0.096
ViewDesignHistory	120	0.003
Reset	501	0.014
Idle	1648	0.046
Disengaged	213	0.006

We used the same method to determine the threshold of `Idle` and `Disengaged` actions in the engineering design activity. Similar to the computational model-building activity, the time distributions were also skewed. The resultant Idle and Disengaged thresholds were 69 seconds and 7 minutes (at the 95^{th} and 99^{th} percentile). Table 7.7 lists the descriptive statistics of the time intervals in the engineering design activity.

Table 7.7: Descriptive statistics of the time intervals during the engineering design activity (total number of time intervals = 35,549)

	Interval
mean	0:00:20.52
std	0:01:34.48
min	0:00:00.05
25%	0:00:01.90
50%	0:00:03.42
75%	0:00:07.05
Idle threshold (95%)	0:01:08.99
Disengaged threshold (99%)	0:06:59.22
max	0:39:45.21

Figure 7.4 provides a visualization of the first-order Markov Chain model derived from all students' activities with interpolated `Idle` and `SoD` actions. Table 7.8 shows the transition probabilities between the state space defined as the engineering design actions (c.f. Section 4.6).

Table 7.8: Transition probabilities of students' computational engineering design processes

	AddMaterial	RemoveMaterial	Reset	RunSimulation	ViewHistory	Idle	SoD
AddMaterial	**0.62**	**0.25**	0.00	0.10	0.00	0.02	0.01
RemoveMaterial	**0.49**	**0.48**	0.01	0.01	0.00	0.01	0.00
Reset	**0.55**	0.00	**0.23**	0.06	0.01	0.08	0.05
RunSimulation	0.04	**0.35**	0.03	**0.27**	0.00	0.20	0.09
ViewDesignHistory	0.13	0.00	0.03	0.11	**0.49**	0.17	0.08
Idle	0.17	**0.42**	0.05	**0.35**	0.01	n/a	n/a
SoD	0.12	**0.46**	0.06	**0.34**	0.02	n/a	n/a

As the two most frequent actions, `AddMaterial` (49%) and `RemoveMaterial` (34%) were two *sink* states in Figure 7.4, and most of the other states have high transition probabilities to these two actions. This result reflects the learning activity in the engineering design of the WRC: students needed to search for satisfying schoolyard designs from a vast problem space of 6^{12} possible solutions.

Similar to the computational modeling activity, students took very short gaps between actions in the engineering design activity and the gaps were even shorter compared to the 25th, 50th, and 75th percentile in Table 7.1. Overall, in the engineering design activity, the

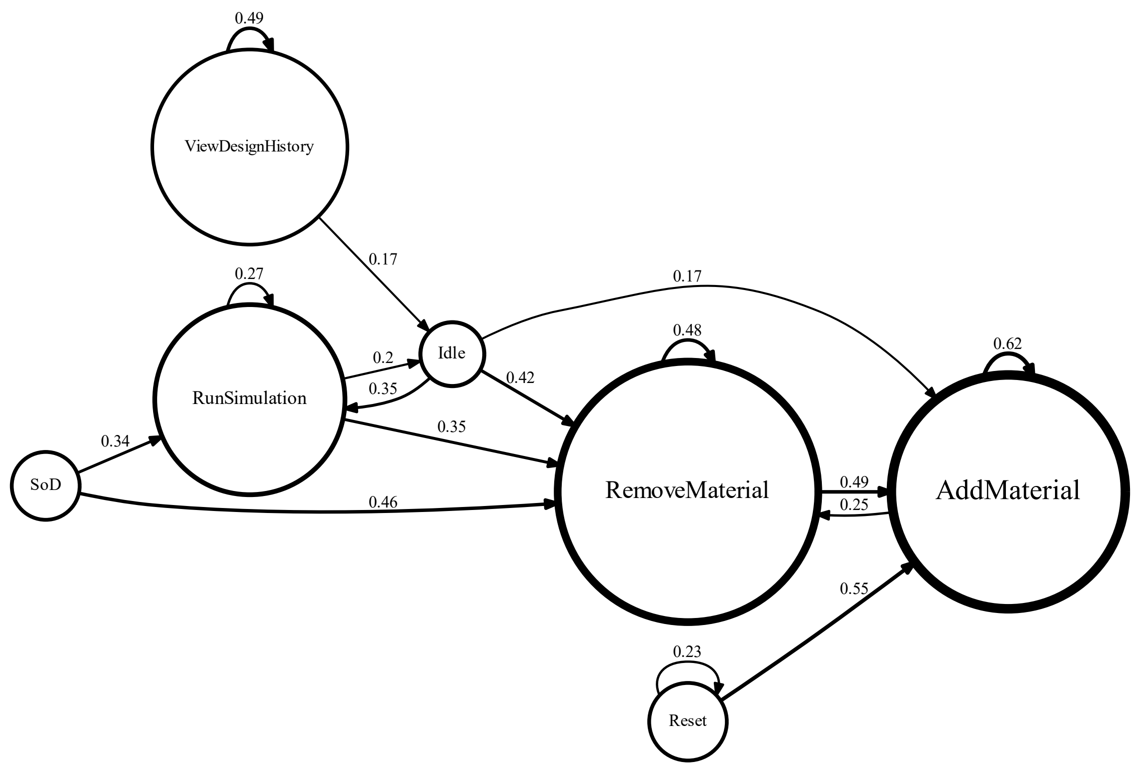


Figure 7.4: Discrete-time Markov chain model of students' engineering design processes

students did not spend much time on applying cognitive and metacognitive strategies that included reflection and planning.

7.2.1 Linking Engineering Design Behaviors to Learning Performance

Using the same the method used to analyze students' strategy use in Section 7.1.2, we analyzed the students' learning behaviors in the engineering design activity to investigate RQ 4, i.e., *How do students utilize strategies to facilitate their learning processes?* After generating MC models for each individual student's engineering design actions, we generated the likelihoods of pairs of actions and applied multivariate regression to examine if these likelihoods were statistically significant predictors of the learning performances. Table 7.9 lists the significant predictors and their standardized regression coefficients (β) ranked in descending order. All of the eight action sequences have high lifts.

Table 7.9: Statistically significant predictors (likelihoods of pairs of actions) of the learning performance

	action sequence	β	*std err*	*p*-value	lift
1	Reset→AddMaterial	0.45	0.20	0.030	1.13
2	SoD→ViewDesignHistory	0.42	0.20	0.047	6.06
3	Idle→RunSimulation	0.37	0.18	0.048	3.58
4	AddMaterial→AddMaterial	-0.31	0.15	0.035	1.25
5	RunSimulation→Disengaged	-0.36	0.18	0.049	6.11
6	ViewDesignHistory→RunSimulation	-0.57	0.24	0.021	1.12

To begin with, sequence #1 involved the `Reset` action, which only accounted for 1.4% of the total number of actions. This sequence was associated with starting over with a schoolyard design from scratch. Although resetting the existing design is task-specific to the WRC and thus difficult to be categorized as a task-general learning strategy to facilitate the learning performance, as illustrated in Figure 7.5, these likelihoods reflected the effort of exploring in the *problem space* (Klahr and Dunbar, 1988), whose contribution to the learning performance had been discovered and discussed in Section 6.5.2 of this dissertation.

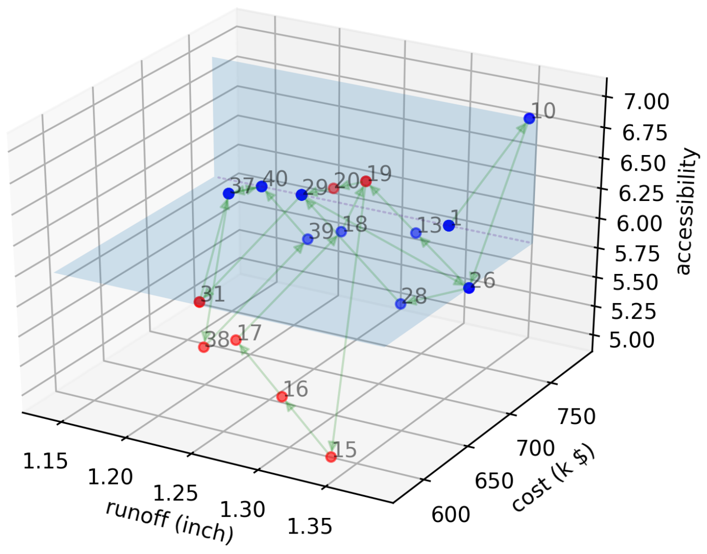


Figure 7.5: Visualization of a student's engineering design process

Second, sequences #2 and #6 both involved with the `ViewDesignHistory` action. Sequence #2 (`SoD` → `ViewDesignHistory`) had both a positive association with the learning performance and a very high lift measure. On the other hand, the likelihood of sequence #6 (`ViewDesignHistory` → `RunSimulation`) was the strongest negative predictor of learning performance. In addition to being strong predictors of the learning performance, these two sequences also reflected the students' strategy use. `ViewDesignHistory` is an action with the lowest total number of instances ($n = 120$) and taken by very few students. As introduced in Section 4.2, clicking on the *DesignHistory* block will pop out a well-formatted table that lists the entire history of all unique schoolyard designs of a student, including the timestamp, the cost, the composition of the materials, and the values of the input and output variables (total rainfall, total absorption, and total runoff). This DSML block is designed specifically to support students' engineering design and problem-solving processes. During the classroom study of the WRC, we witnessed some students' strategic use of this function to optimize their schoolyard designs. For example, as indicated by sequence

#2, some high performing students invoked the reflection metacognitive strategy as they performed the `ViewDesignHistory` action as the first thing to do in a class after coming back from the previous day working on the engineering design activity. In addition, one group of students also used the information from the design history table to prepare for the final presentation in the WRC (e.g., Figure 5.4 in Section 5.1). On the other hand, action sequences like `ViewDesignHistory`→`RunSimulation` suggest non-strategic use of the `ViewDesignHistory` action because viewing the table does not incur any changes to the schoolyard design, and the succeeding *RunSimulation* action would not be informative or meaningful as the design was not modified.

Third, sequence #3 (`Idle`→`RunSimulation`) had a positive association with the learning performance as this situation involved *metacognitive strategies*: if the students are more likely to plan or reflect before running a simulation, their learning performance might be higher than those who run a simulation immediately after committing an explicit action as nonstrategic *trial-and-error* or *tinkering* behaviors.

Fourth, sequence #4 indicated some *depth-first* behaviors (Zhang et al., nd) that had a negative association with the learning performance ($\beta = -0.31$). This action sequence itself is not the problem: overall 49% of the student actions were *AddMaterial* and this pattern could be found in all 99 students using Sequential Pattern Mining and variations despite their learning performance (Zaki, 2001; Kinnebrew et al., 2013). On the other hand, the negative link of the likelihood of this sequence to the learning performance indicates that the low-performing students tended to have overall depth-first behaviors instead of working on a small section of the schoolyard and then testing the quality of the design.

Finally, sequence #5 (`RunSimulation`→`Disengaged`) revealed a behavior that instead of using the simulation results to improve the engineering design, the students became disengaged (not doing anything for at least seven minutes). The likelihood of this behavior was a significant predictor of the learning performance during the computational model-building activity as well, and in both cases, a negative link between students' disengagement and the learning performance was found (c.f. row 9 on Table 7.4).

7.2.2 Differences in the MC Model

After examining pairs of actions in the engineering design activity, in this section, we investigate the students' strategy use during the engineering design activity using longer action sequences (triplets). Similar to the analysis reported in Section 7.1.3, students were divided into the high-performing group (HG) and the low-performing group (LG) using a median-cut of their total post-test scores. Figure 7.6 shows the MC models of all HG and LG students. Unlike the computational modeling task, the structure of the MC models did not differ greatly from HG and LG students.

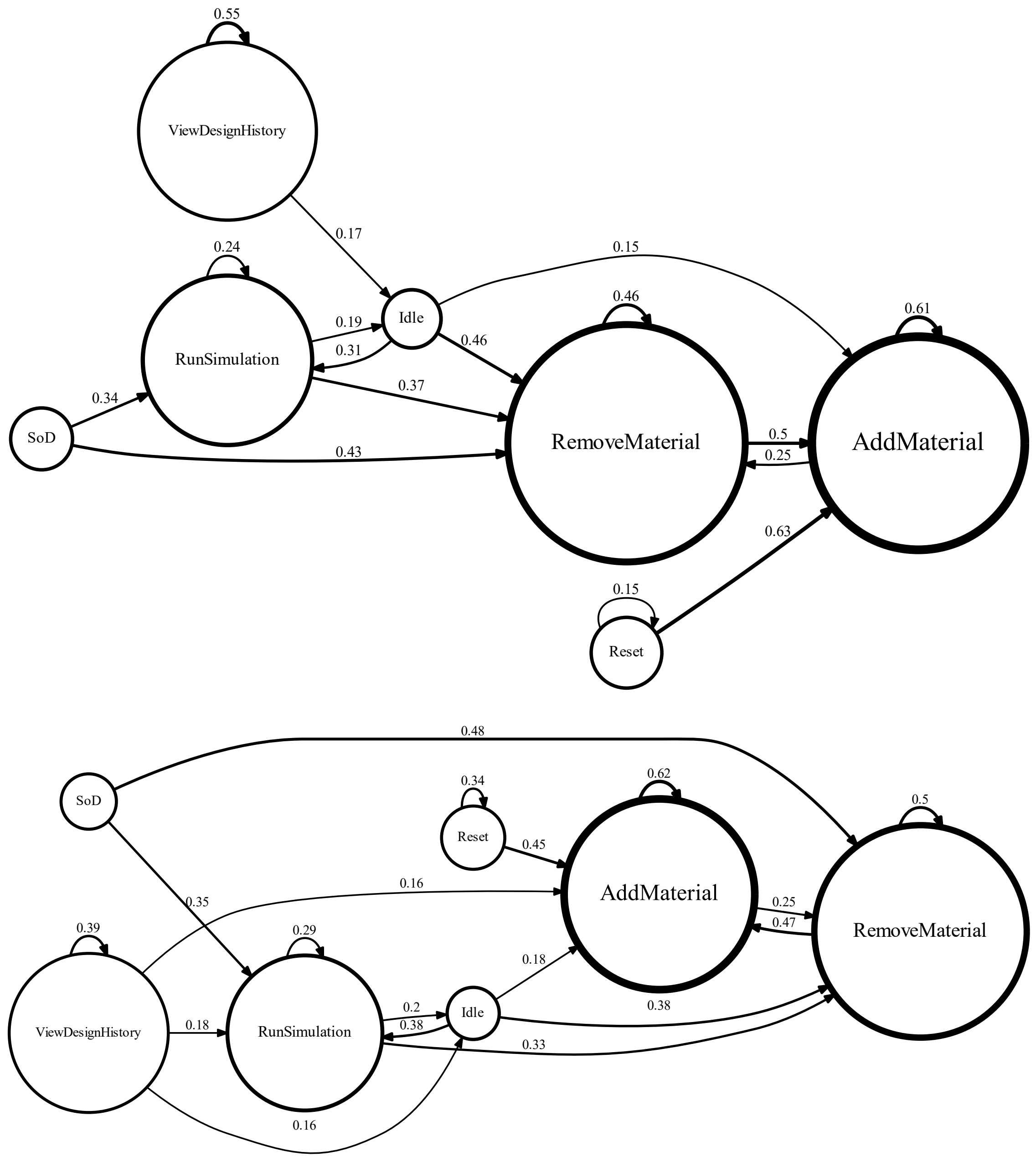


Figure 7.6: The Markov chain models of the engineering processes of the HG (upper image) and LG (lower image) students

Table 7.10: Log-likelihood distributions (median and standard deviations of the log-Likelihood) of high- and low-performing students and Mann-Whitney U-test results

	action sequences	HG (*Median, SD*)	LG (*Median, SD*)	diff	p-value	effect size	lift
1	Idle→AddMaterial→Idle	-9.43 (12.05)	-8.77 (8.32)	-0.2	0.008	0.35	1.22
2	AddMaterial→Idle→AddMaterial	-6.91 (12.01)	-6.03 (8.39)	-1.6	0.009	0.35	0.36
3	AddMaterial→AddMaterial→Idle	-5.26 (5.36)	-4.8 (3.37)	-4.3	0.012	0.35	0.50
4	Idle→AddMaterial→RemoveMaterial	-7.03 (9.63)	-6.49 (7.62)	-1.2	0.021	0.37	0.72
5	AddMaterial→Idle→RunSimulation	-5.8 (9.63)	-5.29 (6.29)	-3.5	0.024	0.37	3.95
6	Reset→AddMaterial→RunSimulation	-7.73 (10.23)	-27.98(12.04)	0.8	0.028	0.63	0.15
7	Idle→RemoveMaterial→RunSimulation	-8.8 (10.14)	-9.87 (11.72)	-0.1	0.032	0.63	0.09
8	RunSimulation→RemoveMaterial→AddMaterial	-4.35 (4.71)	-4.56 (7.83)	-16.0	0.039	0.62	1.23
9	Idle→AddMaterial→AddMaterial	-6.1 (9.64)	-5.43 (7.74)	-3.2	0.046	0.38	1.32
10	RunSimulation→RemoveMaterial→RunSimulation	-7.93 (12.55)	-9.05 (14.82)	4.5	0.048	0.62	0.18

Table 7.10 lists the action sequences whose likelihood among HG and LG students were statistically significant as evaluated by a Mann-Whitney U-test. The lift measurements of the sequences and the effect size of the difference were also reported in Table 7.10. We examined students' strategies reflected from these action sequences of *fair test* and *reflection and planning* in the following subsections.

7.2.2.1 Fair Test Strategy

Sequence #6 presents a situation where a simulation was run with only one new surface material added to an empty schoolyard design. This reflects the case of isolating a single material to test its influence on reducing the runoff. It is notable that sequence #6 was almost impossible for LG students to perform, and the median log-likelihood of -27.98 was due to the additive smoothing. Meanwhile, sequence #10 represents a complement of sequence #6 that the effect of removing a material was examined in isolation between two simulations. The HG students also had a significantly higher likelihood to perform sequence #10 than the LG students with moderate effect sizes. Both sequences had very low lift measures (0.15 and 0.18, respectively). According to the definition of a lift measure (c.f., Section 5.4), their rule bodies (`Reset→AddMaterial` and `RunSimulation→RemoveMaterial`) had a negative effect on the occurrence of the rule head (`RunSimulation`), which made these two sequences unique.

More importantly, these two action sequences reflect a *fair test* strategy with which students make the smallest unit of change to the design and then compare the difference (an empty null design *vs.* a one-material design in the case of `Reset`→`AddMaterial`→`Run Simulation` and any arbitrary design *vs.* one material removed from it in the case of `RunSimulation`→ `RemoveMaterial`→`Run Simulation`). Notably, the concept of conducting *fair tests* was an important aspect thoroughly discussed in the WRC and was also the theme of lesson 12 (c.f. Table 5.1). Students' application of the *fair test* strategy seemed to influence their learning performance as well.

7.2.2.2 Reflection and Planning

Sequences #1, #2, #3, #4, and #9 all consisted of an `Idle` action and two other actions changing the schoolyard design. More specifically, the `Idle`→`AddMaterial` sub-rule existed in all of them. The lift measure for the sub-rule was 0.32 and the cosine measure was 0.048[3], which is much smaller than 0.65 threshold to be considered as frequent co-occurring rules (Merceron and Yacef, 2008).

On the other hand, unlike the `Idle` behaviors that can be identified as reflection or strategic planning during the computational model-building activity, as these sequences do not involve a `RunSimulation` or `ViewDesignHistory` action that were important indicators of applying metacognitive strategies (c.f., Section 7.2.1). As a contrasting case, HG students had a higher likelihood to perform sequence #7 (`Idle`→`RemoveMaterial`→ `RunSimulation`). As a result, the likelihood for HG students to perform these action sequences that indicate a lack of metacognition was significantly lower than the LG students.

7.3 Summary of Chapter

In this Chapter, we linked students' learning behaviors in the computational modeling-building and engineering design activities to their learning performances measured with the post-test scores to investigate RQ 4 of this dissertation, i.e., *How do students utilize strate-*

[3]A cosine measurement can value between 0 and 1. The measurement is calculated as $\frac{P(A \wedge B)}{sqrt(P(A)P(B))}$.

gies to facilitate their learning processes (i.e., the construction of computational models and generating engineering designs)?. Using a data-driven method, we interpolate the implicit actions of idling and disengagement to the students' action sequence and derived Markov Chain models for each individual student.

This work is distinct from recent related research, which has also applied data-driven approaches to study strategy use from traces of learning activities (e.g., Gasevic et al. (2017); Fincham et al. (2018); Whitelock-Wainwright et al. (2020); Matcha et al. (2019); Ahmad Uzir et al. (2020)). The focus of previous work has been on studying undergraduate students' strategy use when video-watching in the context of flipped classrooms (Gasevic et al., 2017; Fincham et al., 2018; Ahmad Uzir et al., 2020), information processing using search tools and a concept map to support essay writing (Whitelock-Wainwright et al., 2020), and learning software programming from MOOCs (Matcha et al., 2019)).

On the other hand, our focus is on studying middle school students' use of strategies. Unlike undergraduate students, middle school students are novice learners who are just beginning to learn complex science and computational concepts. In addition, building and analyzing computational models in science domains by combining information acquisition (learning concepts and practices), computational model building, and model checking is a complex and involved task, and it is interesting to study how these students develop and apply strategies to succeed in their tasks.

By checking the associations between the likelihoods of certain sequence patterns and the learning performance, we identified optimal and sub-optimal behavior patterns that were indicative of learning strategies such as **Reusing Block**, **Reflection and Planning**, **Build then Test**, and **Fair Test**. By extending the short behavior patterns to longer action sequences, we found a significant difference in the use of strategies among high- and low-performing students.

For example, the high-performing students were better at debugging their models by detecting errors in their models after running simulations using the **Build then Test** strategy. Furthermore, high-performers were able to pause, review, and plan while engaging in the model-building tasks. On the other hand, low performing students were more likely to conduct the less effective **trial and error** behaviors.

On the other hand, the type and number of action sequences in which high- and low-performing students significantly differed were affected by (1) the granularity of the definition and recording of the student behaviors and (2) students' variations in performing the learning activities. For example, about 93.1% of the actions during the engineering design task were either `AddMaterial`, `RemoveMaterial`, or `RunSimulation`. As a result, it was more difficult to find variations of the action patterns among HG and LG students compared to the computational model-building task which was more complex and challenging.

Nevertheless, the current work shows that similar to sequence mining approaches e.g., (Agrawal and Srikant, 1995), the pattern mining approach with Markov Chain Modeling in the present work was able to find statistically relevant patterns across representative students' sequence of actions performed in the WRC learning environment both at the student level and at the aggregated level.

Chapter 8

Conclusions

8.1 Contributions

As part of this dissertation research, we have developed a computational model-building and engineering design environment that supports the Water Runoff Challenge. This is one of the first instances of NGSS-aligned curricula that provide an integrated learning framework for science, engineering, and CT. Using the learning environment, sixth-grade students constructed scientific models of the urban runoff scenario and explored the problem space of an engineering design challenge to find an optimal solution to a schoolyard design. This dissertation makes contributions to computer science and learning science research by developing methodologies that address:

1. The complexities of automated evaluation and scoring of student-generated solutions (computational modeling and engineering design) in open-ended learning environments (OELEs).
2. The lack of research investigating the integration of science and engineering facilitated by CT.
3. The insufficient understanding of the affordances and benefits of an integrated curriculum that brings together science, engineering, and CT.
4. The need to develop methods to evaluate and link students' learning and problem-solving processes using state of the art analytic and mining methods.

The research conducted makes contributions to investigating the following research hypotheses: the learning of science and engineering share a synergistic relationship; (2) computational thinking plays a supporting role in the learning of both constructs; (3) understanding students' use of learning strategies offers insights on how to characterize students' successful learning behaviors. We further analyzed these the research hypotheses in terms of four research questions:

1. *How do we design and implement a computational modeling environment to support the integrated learning of science and engineering and how effective is this curriculum in supporting student learning?*

2. *What are the relationships between science learning and engineering performance?*

3. *What is the role of computational thinking in facilitating science learning and engineering design?*

4. *How do students utilize strategies to facilitate their learning processes*

8.1.1 Design and Implementing the OELE

To answer *RQ 1*, this dissertation presented a principled design and implementation approach for an OELE that extends the previous WRC curriculum (c.f., (Chiu et al., 2019)) by enabling computational modeling activities so that students could actively construct computational models of water runoff for different materials using their knowledge of science and CT, instead of just performing the engineering design task with a pre-built model of water runoff. Using a DSML approach, the students' learning activities were tailored to fit their CT, science, and mathematics grade-level abilities.

In addition, students' interaction with the environment was captured and stored as trace data. The trace data collected from a sixth-grade classroom study was then processed and evaluated to produce context-rich measures that describe the learning processes using learning analytics measures. As a result, the behavior and performance measurements calculated from the traces of students' actions in the computational model-building and engineering design activities provided us with a holistic view of the students' learning and problem-solving processes in the WRC curriculum. The results, especially the strong learning gains and the link between students' computational model scores and engineering design performance, demonstrated the benefits of guiding students' computational modeling prior to using the model to solve the engineering problem with this OELE.

8.1.2 Supporting the Integration and the Synergy of Science, Engineering, and CT Learning

Additional results from a classroom study were used to investigate RQ 2 and RQ3. This dissertation demonstrated the instructional benefits of using the WRC and provided empirical evidence to support the integration of engineering activities with science learning and computational model building, especially in early K-12 settings.

Magana and de Jong (2018) summarized three use cases with modeling activities in educational settings (1) learning the practice of modeling itself and integrating system thinking (e.g., (Löhner et al., 2005; Sengupta et al., 2013)), (2) assisting the acquisition of the domain knowledge (e.g., (Mulder et al., 2016); and (3) learning by interacting with ready-made models as virtual labs or simulations, which is the "*most frequent way of using computer models in education*" (Magana and de Jong, 2018, p. 732).

This dissertation showcased a combination of all three use cases suggested by Magana and de Jong (2018) as a way for using computational modeling and CT to integrate science and engineering. This approach merged insights from two learning research traditions: (1) developing computational artifacts and (2) engaging in simulation-based problem-solving. Content-wise, the learning domain focuses on the three-dimensional learning of Earth and Environmental Sciences outlined in the NGSS (National Research Council, 2012) for upper-elementary and lower-middle school students. The design principles and the practical implications of the present work can also be applied to other learning domains for older students such as sustainability, human impact and responsibility to environmental issues, and other aspects of ecology.

8.1.3 Detection and Interpreting Contextualized Learning Strategies

This dissertation also extended the literature on strategy analysis through the modeling, detection, and interpretation of students' learning strategies during the computational model-building and engineering design activities.

From the theoretical perspective, this dissertation provides a contextualized definition of learning strategies that is synthesized from the literature and grounded the definition with the WRC OELE. Learning strategies are defined as conscious and controllable sequences of actions that facilitate and enhance task performance. In the context of the WRC curriculum, students used a wide range of learning strategies to help them construct models of the runoff scenario and then create satisfying schoolyard designs.

From the methodological perspective, the present work demonstrated systematic approaches to model, measure, and evaluate students' learning strategies by (1) extracting action sequences from individual student's trace data; (2) determining and examining the thresholds for implicit *Idle* and *Disengage* behaviors; (3) using Markov Chain models as both *descriptive* and a *generative* model of students' learning processes; (4) generating distributions of the likelihoods of specific action sequences of interest among different student groups using an *ad hoc* treatment allocation (e.g., high- vs. low-performers); and (5) performing hypothesis tests to measure the statistical significance of the difference among the distributions of the strategy use by the two groups. As a result of these analytical approaches, the present work developed methods for (1) extracting students' significant activity patterns; (2) linking them to potential learning strategies; and (3) interpreting strategy use in terms of the students' learning behaviors and performances in the WRC.

The analyses of data collected from the classroom study established that this approach can be applied to detect and identify students' learning strategies in two different learning activities, which addressed RQ 4 in this dissertation research by comparing strategies used by high- and low-performing students.

8.1.4 Other Contributions

Finally, this dissertation makes contributions that advance the understanding of two of the 20 Grand Challenges in Science Education, i.e., (1) Help students explore the personal relevance of science and integrate scientific knowledge into complex practical solutions;

and (2) Create online environments that use stored data from individual students to guide them to virtual experiments that are appropriate for their stage of understanding (Hines et al., 2013, p. 290).

To begin with, the learning activities in the WRC curriculum are personally meaningful to the students, as the learning tasks are situated in an authentic problem that students are familiar with (The Cognition and Technology Group, 1993). Runoff is a common problem in urban areas that causes environmental issues in addition to affecting the usability of school facilities. Therefore, the investigation and design activities in the WRC curriculum are relevant to student's lives.

Second, students' exploration of the runoff *problem space* (Klahr and Dunbar, 1988) is strongly associated with evidence-based reasoning with experimental data to support the understanding of the investigation and design processes. The interactive and data-rich approach to assist student's engineering design activity facilitates the virtual experiments for students to create, record, and revise satisfying schoolyard designs to reduce runoff and prepare for communicating their results to their teachers and classmates.

8.2 Limitations and Future Work

One limitation of the present work is its use of the high- vs. low-performing dichotomy in the investigation of students' learning strategies. The participants were divided into two groups using a median cut of their post-test scores as a common *post hoc* practice in quasi-experimental approaches. However, this treatment had a coarse granularity and may have overlooked some individual characteristics of the learners. To address it, future work can aggregate the students' work at a finer level by creating clusters of learners that show similar characteristics based on learning analytics-based measurements.

This dissertation also creates opportunities for future research in at least four directions: (1) the online prediction of students' learning outcomes and designing adaptive scaffolding, (2) extending the analytical model representations, (3) more in-depth analysis of students' learning and problem-solving processes to derive optimal engineering design solutions, and (4) exploration with alternative methods and evaluating the psychometrics of the assessments. The rest of this section discusses these four aspects in detail.

8.2.1 Prediction, Scaffolding, and Delivery

The present work used multivariate regression models to link the likelihood of certain action sequences to students' learning performance. The purpose of applying this approach does derive the most accurate model that **predicts** of students' learning outcomes while they work on the WRC, which needs to be addressed with additional effort that includes model selection, feature engineering, feature selection, and bias removal. Instead, this approach was used to discover the action sequences that were statistically *indicative* of the learning performance that served as a basis for discovering optimal and sub-optimal learning strategies. Therefore, one direction of future work is to train state-of-the-art predictive models and deploy the models for *online* prediction purposes. To overcome the issue with the relatively small sample size of the training data, Leave-one-out cross validation can be applied to reduce the *generalization error* of the predictive models (Bishop, 2007).

In addition, it was also shown in related work that all students, especially the low performing ones, may benefit from adaptive scaffolding while they are involved in their model-building work (Basu et al., 2017). Therefore, future work can also focus on extending the adaptive learning framework to provide targeted feedback to students when they experience difficulties in their model-building and engineering design tasks. We believe such feedback will help low performing students develop more effective strategies, and, therefore, become better learners and problem solvers.

Furthermore, the data analyses framework for processing the trace data collected during the computational model-building and engineering design activities exists as stand-along software libraries and scripts outside of the NetsBlox environment. To better facilitate the delivery of online feedback and scaffolding, additional software engineering work is required to migrate the data analysis framework into an integrated software infrastructure as a service layer (Papazoglou and Georgakopoulos, 2003).

8.2.2 Extending the AST Representation

In the present work, the abstract syntax tree (AST) representations of the student-generated computational models were mainly used to calculate measurements of the computational model scores. As a benefit of the data logging and processing analyses framework, the contextual information of the model-building processes is preserved for additional analyses (c.f. Section 4.5).

Future work can explore higher-order Markov models to represent the students' learning and problem-solving processes more completely by including more context to the succeeding action. In addition, it can build upon the existing AST representation for studying students' computational modeling processes as an alternative way of the Markov Chain modeling approach employed in the present work. For example, the DSML and primitive computational blocks can be embedded with various dimensions of the domain and CT concepts (e.g., motion, conditional, loops). A model edit action on the AST can then be tagged with the embedded semantics of the computational block to infer which problem-solving sub-task a student is addressing. This approach not only will enable a more contextualized representation of the computational modeling process, but also a way to understand students' problem-solving processes, especially decomposition and modularization, two key practices of computational thinking (Wing, 2011).

8.2.2.1 Systematicity of the Search for Solutions

In the present work, we used a variable *total Euclidean distance* between students' tested schoolyard designs to measure the students' effort and effectiveness of exploring a *problem space* to search for optimal design solutions (Klahr and Dunbar, 1988). In our previous work, we have also created a *testing variability index* that was calculated as the difference between the total Euclidean distance of the first $\frac{1}{3}$ and the last $\frac{1}{3}$ of tested designs of a student's.

Because the index indicates a change in solution approach from early tests to the later ones, we hypothesized that as a way to describe students' **systematicity** of search for an engineering design solution, large testing variability indices would indicate a situation where a student explore a vast *hypothesis space* and then converged to an optimal solution. This idea was supported with evidence of two vignettes as a case study reported in McElhaney et al. (2020) (see Figure 4.14 for the student who had a negative (small) testing variability index and performed poorly in almost all WRC learning tasks). On the other hand, a careful examination of the *testing variability index* as a variable for all 99 students only partially supported the convergence hypothesis.

This problem was partially caused by students' large variations of the number of tests on the schoolyard designs (especially fully implemented designs): if a considerable number of students did not perform a large number of tests, the variability between the first thirds and last thirds of tests become arbitrary. Therefore, the testing variability index was meaningful only for those students who made sufficiently more engineering designs.

As a result, in the present work, the *testing variability index* was excluded from the Path Analysis to study the interconnected variables describing the students' learning of science, engineering, and CT contents (c.f., Section 6.5). On the other hand, future work can delve deep into the **systematicity** of students' search processes of optimal solutions in the problem space with metrics calculated from the existing behavior measures such as the *total Euclidean distance* or the *cosine differences* between consecutive designs, etc. Future work on this line of effort should also consider addressing the issue of variance, especially with standardized or normalized variables.

8.2.3 Structural Equation Modeling and Psychometrics

In the present work, we used Path Analysis (Wright, 1983) instead of Structural Equation Modeling (SEM) (Kline, 2015) to study the relationship between the variables used to measure students' learning behaviors and performances in the computational model-building and engineering design activities. Two of the motivations for choosing the path representation over SEM were (1) because there is a well-defined structure, and the order of the WRC curriculum where each learning activity can be measured explicitly with observable variables; and (2) because we wanted to measure the staged effects of the variables on the learning progression, for example, while controlling the science and CT pre-test scores as covariates, how did students' computational model score impact the quality of the schoolyard design.

On the other hand, SEM methods allow the use of *latent* variables to describe constructs such as students' proficiency and knowledge of science, engineering, and CT. An additional benefit of SEM is that the *loading factors* of assessment items can be used to understand and evaluate psychometrics such as the reliability and validity of the assessment (e.g., whether the engineering design test items have an acceptable internal consistency (Cronbach's α) or whether the CT test items truthfully evaluate students' understanding of CT instead of other constructs). Therefore, future work can explore an array of possible SEM representations and improve the quality of assessment design by also studying psychometrics.

REFERENCES

Agrawal, R. and Srikant, R. (1995). Mining sequential patterns. In *Proceedings of the Eleventh International Conference on Data Engineering*, pages 3–14.

Ahmad Uzir, N., Gašević, D., Matcha, W., Jovanović, J., and Pardo, A. (2020). Analytics of time management strategies in a flipped classroom. *Journal of Computer Assisted Learning*, 36(1):70–88.

Ahn, J. (2002). Beyond single equation regression analysis: Path analysis and multi-stage regression analysis. *American journal of pharmaceutical education*, 66(1):37–41.

Alexander, P. A., Graham, S., and Harris, K. R. (1998). A perspective on strategy research: Progress and prospects. *Educational psychology review*, 10(2):129–154.

Anderson, J. (1983). Cognitive science series. the architecture of cognition.

Azevedo, R., Harley, J., Trevors, G., Duffy, M., Feyzi-Behnagh, R., Bouchet, F., and Landis, R. (2013). Using trace data to examine the complex roles of cognitive, metacognitive, and emotional self-regulatory processes during learning with multi-agent systems. In *International handbook of metacognition and learning technologies*, pages 427–449. Springer.

Azevedo, R., Johnson, A., Chauncey, A., and Burkett, C. (2010). Self-regulated learning with metatutor: Advancing the science of learning with metacognitive tools. In *New science of learning*, pages 225–247. Springer.

Baker, R. S., D'Mello, S. K., Rodrigo, M. M. T., and Graesser, A. C. (2010). Better to be frustrated than bored: The incidence, persistence, and impact of learners' cognitive–affective states during interactions with three different computer-based learning environments. *International Journal of Human-Computer Studies*, 68(4):223–241.

Bannert, M., Reimann, P., and Sonnenberg, C. (2014). Process mining techniques for analysing patterns and strategies in students' self-regulated learning. *Metacognition and learning*, 9(2):161–185.

Baron, J. (1985). *Rationality and Intelligence*. Cambridge University Press.

Barr, D., Harrison, J., and Conery, L. (2011). Computational thinking: A digital age skill for everyone. *Learning & Leading with Technology*, 38(6):20–23.

Barr, V. and Stephenson, C. (2011). Bringing computational thinking to k-12: what is involved and what is the role of the computer science education community? *Acm Inroads*, 2(1):48–54.

Basawapatna, A., Koh, K. H., Repenning, A., Webb, D. C., and Marshall, K. S. (2011). Recognizing computational thinking patterns. In *Proceedings of the 42nd ACM technical symposium on Computer science education*, pages 245–250.

Basu, S. and Biswas, G. (2016). Providing adaptive scaffolds and measuring their effectiveness in open ended learning environments.

Basu, S., Biswas, G., and Kinnebrew, J. S. (2016a). Using multiple representations to simultaneously learn computational thinking and middle school science.

Basu, S., Biswas, G., and Kinnebrew, J. S. (2017). Learner modeling for adaptive scaffolding in a computational thinking-based science learning environment. *User Modeling and User-Adapted Interaction*, 27(1):5–53.

Basu, S., Biswas, G., Sengupta, P., Dickes, A., Kinnebrew, J. S., and Clark, D. (2016b). Identifying middle school students' challenges in computational thinking-based science learning. *Research and practice in technology enhanced learning*, 11(1):13.

Basu, S., Dickes, A., Kinnebrew, J. S., Sengupta, P., and Biswas, G. (2013). Ctsim: A computational thinking environment for learning science through simulation and modeling. In *CSEDU*, pages 369–378.

Basu, S., Dukeman, A., Kinnebrew, J. S., Biswas, G., and Sengupta, P. (2014). Investigating student generated computational models of science.

Basu, S., Sengupta, P., and Biswas, G. (2015). A scaffolding framework to support learning of emergent phenomena using multi-agent-based simulation environments. *Research in Science Education*, 45(2):293–324.

Bille, P. (2005). A survey on tree edit distance and related problems. *Theoretical computer science*, 337(1-3):217–239.

Birrell, A. and Nelson, B. J. (1984). Implementing remote procedure calls. *ACM Trans. Comput. Syst.*, 2(1):39–59.

Bishop, C. M. (2007). *Pattern recognition and machine learning, 5th Edition*. Information science and statistics. Springer.

Biswas, G., Leelawong, K., Schwartz, D., Vye, N., and at Vanderbilt, T. T. A. G. (2005). Learning by teaching: A new agent paradigm for educational software. *Applied Artificial Intelligence*, 19(3-4):363–392.

Biswas, G., Segedy, J. R., and Bunchongchit, K. (2016). From design to implementation to practice a learning by teaching system: Betty's brain. *International Journal of Artificial Intelligence in Education*, 26(1):350–364.

Blikstein, P. (2013). Digital fabrication and 'making'in education: The democratization of invention. *FabLabs: Of machines, makers and inventors*, 4(1):1–21.

Boekaerts, M. (1996). Self-regulated learning at the junction of cognition and motivation. *European psychologist*, 1(2):100.

Brady, C., Orton, K., Weintrop, D., Anton, G., Rodriguez, S., and Wilensky, U. (2016). All roads lead to computing: Making, participatory simulations, and social computing as pathways to computer science. *IEEE Transactions on Education*, 60(1):59–66.

Brennan, K. and Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. In *Proceedings of the 2012 annual meeting of the American educational research association, Vancouver, Canada*, volume 1, page 25.

Broll, B. (2018). *Collaborative Educational Environment Design for Accessible Distributed Computing*. Vanderbilt University.

Broll, B., Lédeczi, Á., Zare, H., Do, D. N., Sallai, J., Völgyesi, P., Maróti, M., Brown, L., and Vanags, C. (2018). A visual programming environment for introducing distributed computing to secondary education. *Journal of Parallel and Distributed Computing*, 118:189–200.

Brophy, S., Klein, S., Portsmore, M., and Rogers, C. (2008). Advancing engineering education in p-12 classrooms. *Journal of Engineering Education*, 97(3):369–387.

Browne, J. C. (1986). Framework for formulation and analysis of parallel computation structures. *Parallel Computing*, 3(1):1–9.

Cheng, B., Ructtinger, L., Fujii, R., and Mislevy, R. (2010). Assessing systems thinking and complexity in science (large-scale assessment technical report 7). *Menlo Park, CA: SRI International. Downloaded September*, 11:2010.

Chi, M. T. (2005). Commonsense conceptions of emergent processes: Why some misconceptions are robust. *The journal of the learning sciences*, 14(2):161–199.

Chi, M. T., Roscoe, R. D., Slotta, J. D., Roy, M., and Chase, C. C. (2012). Misconceived causal explanations for emergent processes. *Cognitive science*, 36(1):1–61.

Chin, D. B., Dohmen, I. M., Cheng, B. H., Oppezzo, M. A., Chase, C. C., and Schwartz, D. L. (2010). Preparing students for future learning with teachable agents. *Educational Technology Research and Development*, 58(6):649–669.

Chiu, J., McElhaney, K. W., Zhang, N., Biswas, G., Fried, R., Basu, S., and Alozie, N. (2019). A principled approach to ngss-aligned curriculum development integrating science, engineering, and computation: A pilot study. Paper presented at the 2019 NARST Annual International Conference.

Collins, A. and Ferguson, W. (1993). Epistemic forms and epistemic games: Structures and strategies to guide inquiry. *Educational psychologist*, 28(1):25–42.

Cornford, I. R. (2002). Learning-to-learn strategies as a basis for effective lifelong learning. *International journal of lifelong education*, 21(4):357–368.

Cortez, P. and Silva, A. M. G. (2008). Using data mining to predict secondary school student performance.

Cunningham, C. M. (2017). *Engineering in elementary STEM education: Curriculum design, instruction, learning, and assessment*. Teachers College Press.

Cunningham, C. M., Knight, M. T., Carlsen, W. S., and Kelly, G. (2007). Integrating engineering in middle and high school classrooms. *International Journal of Engineering Education*, 23(1):3.

de Jong, T., Linn, M. C., and Zacharia, Z. C. (2013). Physical and virtual laboratories in science and engineering education.

Denning, P. J. (2009). The profession of it beyond computational thinking. *Communications of the ACM*, 52(6):28–30.

Dent, E. B. (1999). Complexity science: A worldview shift. *Emergence*, 1(4):5–19.

Derry, S. J. (1990). Learning strategies for acquiring useful knowledge. *Dimensions of thinking and cognitive instruction*, pages 347–379.

Donker, A., de Boer, H., Kostons, D., van Ewijk, C. D., and van der Werf, M. (2014). Effectiveness of learning strategy instruction on academic performance: A meta-analysis. *Educational Research Review*, 11:1–26.

Du Boulay, B. and Luckin, R. (2001). Modelling human teaching tactics and strategies for tutoring systems. *International Journal of Artificial Intelligence in Education*, 12(3):235–256.

English, L. D., Hudson, P. B., and Dawes, L. A. (2011). Middle school students' perceptions of engineering. In *STEM in education conference: science, technology, engineering and mathematics in education conference*. Queensland University of Technology.

Fincham, E., Gašević, D., Jovanović, J., and Pardo, A. (2018). From study tactics to learning strategies: an analytical method for extracting interpretable representations. *IEEE Transactions on Learning Technologies*, 12(1):59–72.

Fitts, P. M. (1962). Factors in complex skill training. *Training research and education*, pages 177–197.

Flavell, J. H. (1979). Metacognition and cognitive monitoring: A new area of cognitive–developmental inquiry. *American psychologist*, 34(10):906.

Fortus, D., Krajcik, J., Dershimer, R. C., Marx, R. W., and Mamlok-Naaman, R. (2005). Design-based science and real-world problem-solving. *International Journal of Science Education*, 27(7):855–879.

Fralick, B., Kearn, J., Thompson, S., and Lyons, J. (2009). How middle schoolers draw engineers and scientists. *Journal of Science Education and Technology*, 18(1):60–73.

Gansner, E. R. and North, S. C. (2000). An open graph visualization system and its applications to software engineering. *SOFTWARE - PRACTICE AND EXPERIENCE*, 30(11):1203–1233.

Garner, R. (1988). 5 - verbal-report data on cognitive and metacognitive strategies. In Weinstein, C. E., Goetz, E. T., and Alexander, P. A., editors, *Learning and Study Strategies*, pages 63–76. Academic Press.

Gasevic, D., Jovanovic, J., Pardo, A., and Dawson, S. (2017). Detecting learning strategies with analytics: Links with self-reported measures and academic performance. *Journal of Learning Analytics*, 4(2):113–128.

George, R. (2006). A cross-domain analysis of change in students' attitudes toward science and attitudes about the utility of science. *International journal of science education*, 28(6):571–589.

Gilbert, J. K., Boulter, C., and Rutherford, M. (1998). Models in explanations, part 1: Horses for courses? *International Journal of Science Education*, 20(1):83–97.

Glickman, M. E., Rao, S. R., and Schultz, M. R. (2014). False discovery rate control is a recommended alternative to bonferroni-type adjustments in health studies. *Journal of clinical epidemiology*, 67(8):850–857.

Grover, S. and Pea, R. (2013). Computational thinking in k–12: A review of the state of the field. *Educational researcher*, 42(1):38–43.

Grover, S., Pea, R., and Cooper, S. (2015). Systems of assessments" for deeper learning of computational thinking in k-12. In *Proceedings of the 2015 annual meeting of the American educational research association*, pages 15–20.

Hadwin, A. F., Nesbit, J. C., Jamieson-Noel, D., Code, J., and Winne, P. H. (2007). Examining trace data to explore self-regulated learning. *Metacognition and Learning*, 2(2-3):107–124.

Hannafin, M., Land, S., and Oliver, K. (1999). Open learning environments: Foundations, methods, and models. *Instructional-design theories and models: A new paradigm of instructional theory*, 2:115–140.

Hannafin, M. J., Hill, J. R., Land, S. M., and Lee, E. (2014). *Student-Centered, Open Learning Environments: Research, Theory, and Practice*, chapter 51, pages 641–651. Springer New York.

Harris, A., Bonnett, V., Luckin, R., Yuill, N., and Avramides, K. (2009). Scaffolding effective help-seeking behaviour in mastery and performance oriented learners. In *AIED*, pages 425–432.

Harrison, A. G. and Treagust, D. F. (1996). Secondary students' mental models of atoms and molecules: Implications for teaching chemistry. *Science education*, 80(5):509–534.

Harrison, A. G. and Treagust, D. F. (2000). A typology of school science models. *International journal of science education*, 22(9):1011–1026.

Hines, P. J., Mervis, J., McCartney, M., and Wible, B. (2013). Grand challenges in science education. Plenty of challenges for all. Introduction. *Science (New York, N.Y.)*, 340(6130):290–291.

Hirsch, L. S., Carpinelli, J. D., Kimmel, H., Rockland, R., and Bloom, J. (2007). The differential effects of pre-engineering curricula on middle school students' attitudes to and knowledge of engineering careers. In *2007 37th Annual Frontiers In Education Conference-Global Engineering: Knowledge Without Borders, Opportunities Without Passports*, pages S2B–17. IEEE.

Hmelo, C. E., Holton, D. L., and Kolodner, J. L. (2000). Designing to learn about complex systems. *The Journal of the Learning Sciences*, 9(3):247–298.

Hogan, K. and Thomas, D. (2001). Cognitive comparisons of students' systems modeling in ecology. *Journal of Science Education and Technology*, 10(4):319–345.

Hopper, M. and Stave, K. A. (2008). Assessing the effectiveness of systems thinking interventions in the classroom. In *26th International Conference of the System Dynamics Society*.

Hutchins, N., Biswas, G., Grover, S., Basu, S., and Snyder, C. (2019a). A systematic approach for analyzing students' computational modeling processes in c2stem. In *International Conference on Artificial Intelligence in Education*, pages 116–121. Springer.

Hutchins, N., Biswas, G., Maroti, M., Ledezci, A., and Broll, B. (2018). A design-based approach to a classroom-centered oele. In *International conference on artificial intelligence in education*, pages 155–159. Springer.

Hutchins, N., Biswas, G., Zhang, N., Snyder, C., Lédeczi, Á., and Maróti, M. (n.d.). Domain-specific modeling languages in computer-based learning environments: A systematic approach to scaffold science learning through computational modeling. under review.

Hutchins, N., Shi, C., and Biswas, G. (2019b). A high school computational modeling approach to studying the effects of climate change on coral reefs. In *Annual Meeting of the American Education Research Association*.

Hutchins, N. M., Biswas, G., Maróti, M., Lédeczi, Á., Grover, S., Wolf, R., Blair, K. P., Chin, D., Conlin, L., Basu, S., and McElhaney, K. (2020). C2stem: a system for synergistic learning of physics and computational thinking. *Journal of Science Education and Technology*, pages 83–100.

Irgens, G. A., Dabholkar, S., Bain, C., Woods, P., Hall, K., Swanson, H., Horn, M., and Wilensky, U. (2020). Modeling and measuring high school students' computational thinking practices in science. *Journal of Science Education and Technology*, 29(1):137–161.

ISTE and CSTA (2016). Operational definition of computational thinking.

Jacobson, M. J. and Wilensky, U. (2006). Complex systems in education: Scientific and educational importance and implications for the learning sciences. *The Journal of the learning sciences*, 15(1):11–34.

Jona, K., Wilensky, U., Trouille, L., Horn, M., Orton, K., Weintrop, D., and Beheshti, E. (2014). Embedding computational thinking in science, technology, engineering, and math (ct-stem). In *future directions in computer science education summit meeting, Orlando, FL.*

Kafai, Y. B. and Ching, C. C. (2001). Affordances of collaborative software design planning for elementary students' science talk. *The Journal of the Learning Sciences*, 10(3):323–363.

Karmiloff-Smith, A. and Inhelder, B. (1974). If you want to get ahead, get a theory. *Cognition*, 3(3):195–212.

Kinnebrew, J. S., Loretz, K. M., and Biswas, G. (2013). A contextualized, differential sequence mining method to derive students' learning behavior patterns. *JEDM| Journal of Educational Data Mining*, 5(1):190–219.

Kinnebrew, J. S., Segedy, J. R., and Biswas, G. (2014). Analyzing the temporal evolution of students' behaviors in open-ended learning environments. *Metacognition and learning*, 9(2):187–215.

Kinnebrew, J. S., Segedy, J. R., and Biswas, G. (2017). Integrating model-driven and data-driven techniques for analyzing learning behaviors in open-ended learning environments. *IEEE Transactions on Learning Technologies*, 10(2):140–153.

Kirby, J. R. (1988). Style, strategy, and skill in reading. In *Learning strategies and learning styles*, pages 229–274. Springer.

Klahr, D. and Dunbar, K. (1988). Dual space search during scientific reasoning. *Cognitive science*, 12(1):1–48.

Kline, R. B. (2015). *Principles and practice of structural equation modeling*. Guilford publications.

Koedinger, K. R., Alibali, M. W., and Nathan, M. J. (2008). Trade-offs between grounded and abstract representations: Evidence from algebra problem solving. *Cognitive Science*, 32(2):366–397.

Kokoska, S. and Zwillinger, D. (2000). *CRC standard probability and statistics tables and formulae*. CRC Press.

Kolodner, J., Crismond, D., Fasse, B., Gray, J., Holbrook, J., and Puntembakar, S. (2003a). Putting a student-centered learning by design™ curriculum into practice: Lessons learned. *Journal of the Learning Sciences*, 12(4):495–547.

Kolodner, J. L., Camp, P. J., Crismond, D., Fasse, B., Gray, J., Holbrook, J., Puntambekar, S., and Ryan, M. (2003b). Problem-based learning meets case-based reasoning in the middle-school science classroom: Putting learning by design (tm) into practice. *The journal of the learning sciences*, 12(4):495–547.

Kolodner, J. L., Crismond, D., Gray, J., Holbrook, J., and Puntambekar, S. (1998). Learning by design from theory to practice. In *Proceedings of the international conference of the learning sciences*, volume 98, pages 16–22.

Land, S. M. (2000). Cognitive requirements for learning with open-ended learning environments. *Educational Technology Research and Development*, 48(3):61–78.

Leelawong, K. and Biswas, G. (2008). Designing learning by teaching agents: The betty's brain system. *International Journal of Artificial Intelligence in Education*, 18(3):181–208.

Lehrer, R. and Schauble, L. (2006). *Cultivating model-based reasoning in science education.* Cambridge University Press.

Lesh, R., Doerr, H. M., Carmona, G., and Hjalmarson, M. (2003). Beyond constructivism. *Mathematical Thinking and Learning*, 5(2-3):211–233.

Lilliefors, H. W. (1967). On the kolmogorov-smirnov test for normality with mean and variance unknown. *Journal of the American statistical Association*, 62(318):399–402.

Löhner, S., van Joolingen, W. R., Savelsbergh, E. R., and van Hout-Wolters, B. (2005). Students' reasoning during modeling in an inquiry learning environment. *Computers in Human Behavior*, 21(3):441–461.

Louca, L. T. and Zacharia, Z. C. (2015). Examining learning through modeling in k-6 science education. *Journal of Science Education and Technology*, 24(2-3):192–215.

Lowry, R. (2014). Concepts and applications of inferential statistics.

Luckin, R. and du Boulay, B. (2016). Reflections on the ecolab and the zone of proximal development. *International Journal of Artificial Intelligence in Education*, 26(1):416–430.

Luckin, R. and Hammerton, L. (2002). Getting to know me: Helping learners understand their own learning needs through metacognitive scaffolding. In *International Conference on Intelligent Tutoring Systems*, pages 759–771. Springer.

Magana, A. J. and de Jong, T. (2018). Modeling and simulation practices in engineering education. *Computer Applications in Engineering Education*, 26(4):731–738.

Maltese, A. V. and Tai, R. H. (2010). Eyeballs in the fridge: Sources of early interest in science. *International Journal of Science Education*, 32(5):669–685.

Matcha, W., Gašević, D., Jovanović, J., Pardo, A., Maldonado-Mahauad, J., and Pérez-Sanagustín, M. (2019). Detection of learning strategies: A comparison of process, sequence and network analytic approaches. In *European Conference on Technology Enhanced Learning*, pages 525–540. Springer.

Mathews, R. C., Buss, R. R., Stanley, W. B., Blanchard-Fields, F., Cho, J. R., and Druhan, B. (1989). Role of implicit and explicit processes in learning from examples: A synergistic effect. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 15(6):1083.

Mayer, R. E. (1988). 2 - learning strategies: An overview. In Weinstein, C. E., Goetz, E. T., and Alexander, P. A., editors, *Learning and Study Strategies*, pages 11–22. Academic Press.

McElhaney, K. W., Basu, S., Wetzel, T., and Boyce, J. (2019). Three-dimensional assessment of ngss upper elementary engineering design performance expectations. Paper presented at the 2019 NARST Annual International Conference.

McElhaney, K. W., Zhang, N., Basu, S., McBride, E., Biswas, G., and Chiu, J. (2020). Using computational modeling to integrate science and engineering curricular activities. Paper presented at the 2020 International Conference of Learning Sciences, Nashville, USA.

Mehalik, M. M., Doppelt, Y., and Schuun, C. D. (2008). Middle-school science through design-based learning versus scripted inquiry: Better overall science concept learning and equity gap reduction. *Journal of Engineering Education*, 97(1):71–85.

Mendez, G. R., du Boulay, B., and Luckin, R. (2005). Be bold and take a challenge": Could motivational strategies improve help-seeking. In *Proceedings of the 2005 conference on Artificial Intelligence in Education: Supporting Learning through Intelligent and Socially Informed Technology*, pages 459–466.

Merceron, A. and Yacef, K. (2008). Interestingness measures for association rules in educational data. In *Educational Data Mining 2008*.

Metcalf, S. J., Krajcik, J., and Soloway, E. (2000). Model-it: A design retrospective. *Innovations in science and mathematics education*, pages 77–115.

Mislevy, R. J., Almond, R. G., and Lukas, J. F. (2003). A brief introduction to evidence-centered design. *ETS Research Report Series*, 2003(1):i–29.

Moore, T. J., Tank, K. M., Glancy, A. W., and Kersten, J. A. (2015). Ngss and the landscape of engineering in k-12 state science standards. *Journal of Research in Science Teaching*, 52(3):296–318.

Mulder, Y. G., Bollen, L., de Jong, T., and Lazonder, A. W. (2016). Scaffolding learning by modelling: The effects of partially worked-out models. *Journal of research in science teaching*, 53(3):502–523.

Munshi, A., Rajendran, R., Ocumpaugh, J., Biswas, G., Baker, R. S., and Paquette, L. (2018). Modeling learners' cognitive and affective states to scaffold srl in open-ended learning environments. In *Proceedings of the 26th Conference on User Modeling, Adaptation and Personalization*, pages 131–138. ACM.

NAE and NASEM (2019). *Science and Engineering for Grades 6-12: Investigation and Design at the Center*. The National Academies Press, Washington, DC.

National Research Council (2006). *America's lab report: Investigations in high school science*. National Academies Press.

National Research Council (2009). *Engineering in K-12 education: Understanding the status and improving the prospects*. National Academies Press.

National Research Council (2012). *A framework for K-12 science education: Practices, crosscutting concepts, and core ideas*. National Academies Press.

NGSS Lead States (2013). Next generation science standards: For states, by states.

Nisbet, J. and Shucksmith, J. (2017). *Learning strategies*. Routledge.

NRC (2000). *How people learn: Brain, mind, experience, and school: Expanded edition*. National Academies Press.

NRC (2010). *Report of a workshop on the scope and nature of computational thinking*. National Academies Press.

NRC (2011). *Report of a workshop on the pedagogical aspects of computational thinking*. National Academies Press.

Ohlsson, S. (1986). Some principles of intelligent tutoring. *Instructional science*, 14(3-4):293–326.

Oxford, R. L. (2011). Strategies for learning a second or foreign language. *Language Teaching*, 44(2):167.

Panadero, E. and Alonso Tapia, J. (2014). How do students self-regulate?: review of zimmerman's cyclical model of self-regulated learning. *Anales de psicologia*.

Panadero, E., Klug, J., and Järvelä, S. (2016). Third wave of measurement in the self-regulated learning field: when measurement and intervention come hand in hand. *Scandinavian Journal of Educational Research*, 60(6):723–735.

Papazoglou, M. P. and Georgakopoulos, D. (2003). Service-oriented computing. *Communications of the ACM*, 46(10):25–28.

Papert, S. (1986). *Constructionism: A new opportunity for elementary science education*. Massachusetts Institute of Technology, Media Laboratory, Epistemology and ....

Papert, S. and Harel, I. (1991). Situating constructionism. *Constructionism*, 36(2):1–11.

Pearl, J. and Mackenzie, D. (2018). *The book of why: the new science of cause and effect*. Basic Books.

Penner, D. E. (2000). Chapter 1: cognition, computers, and synthetic science: building knowledge and meaning through modeling. *Review of research in education*, 25(1):1–35.

Penner, D. E., Giles, N. D., Lehrer, R., and Schauble, L. (1997). Building functional models: Designing an elbow. *Journal of Research in Science Teaching: The Official Journal of the National Association for Research in Science Teaching*, 34(2):125–143.

Penner, D. E., Lehrer, R., and Schauble, L. (1998). From physical models to biomechanics: A design-based modeling approach. *Journal of the Learning Sciences*, 7(3-4):429–449.

Pianta, R. C., Belsky, J., Houts, R., and Morrison, F. (2007). Opportunities to learn in america's elementary classrooms. *Science*, 315(5820):1795–1796.

Pressley, M., Goodchild, F., Fleet, J., Zajchowski, R., and Evans, E. D. (1989). The challenges of classroom strategy instruction. *The Elementary School Journal*, 89(3):301–342.

Rees, A., García-Peñalvo, F. J., Jormanainen, I., Tuul, M., and Reimann, D. (2016). An overview of the most relevant literature on coding and computational thinking with emphasis on the relevant issues for teachers.

Repenning, A., Ioannidou, A., and Zola, J. (2000). Agentsheets: End-user programmable simulations. *Journal of Artificial Societies and Social Simulation*, 3(3):351–358.

Russell, S. J. and Norvig, P. (2003). *Artificial intelligence - a modern approach, 2nd Edition*. Prentice Hall series in artificial intelligence. Prentice Hall.

Sadler, P. M., Sonnert, G., Hazari, Z., and Tai, R. (2012). Stability and volatility of stem career interest in high school: A gender study. *Science education*, 96(3):411–427.

Schauble, L., Klopfer, L. E., and Raghavan, K. (1991). Students' transition from an engineering model to a science model of experimentation. *Journal of Research in Science Teaching*, 28(9):859–882.

Schmeck, R. R. (2013). *Learning strategies and learning styles*. Springer Science & Business Media.

Schraw, G., Crippen, K. J., and Hartley, K. (2006). Promoting self-regulation in science education: Metacognition as part of a broader perspective on learning. *Research in science education*, 36(1-2):111–139.

Schreiber, J. B., Nora, A., Stage, F. K., Barlow, E. A., and King, J. (2006). Reporting structural equation modeling and confirmatory factor analysis results: A review. *The Journal of educational research*, 99(6):323–338.

Schunk, D. and Greene, J. (2018). *Handbook of Self-Regulation of Learning and Performance*. Routledge.

Schwarz, C. V., Reiser, B. J., Davis, E. A., Kenyon, L., Achér, A., Fortus, D., Shwartz, Y., Hug, B., and Krajcik, J. (2009). Developing a learning progression for scientific modeling: Making scientific modeling accessible and meaningful for learners. *Journal of Research in Science Teaching: The Official Journal of the National Association for Research in Science Teaching*, 46(6):632–654.

Schwarz, C. V. and White, B. Y. (2005). Metamodeling knowledge: Developing students' understanding of scientific modeling. *Cognition and instruction*, 23(2):165–205.

Segedy, J. R., Kinnebrew, J. S., and Biswas, G. (2015a). Coherence over time: understanding day-to-day changes in students' open-ended problem solving behaviors. In *International Conference on Artificial Intelligence in Education*, pages 449–458. Springer.

Segedy, J. R., Kinnebrew, J. S., and Biswas, G. (2015b). Using coherence analysis to characterize self-regulated learning behaviours in open-ended learning environments. *Journal of Learning Analytics*, 2(1):13–48.

Sengupta, P., Dickes, A., Farris, A. V., Karan, A., Martin, D., and Wright, M. (2015). Programming in k-12 science classrooms. *Communications of the ACM*, 58(11):33–35.

Sengupta, P., Kinnebrew, J. S., Basu, S., Biswas, G., and Clark, D. (2013). Integrating computational thinking with k-12 science education using agent-based computation: A theoretical framework. *Education and Information Technologies*, 18(2):351–380.

Shute, V. J., Sun, C., and Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review*, 22:142–158.

Siegal, S. (1956). *Nonparametric statistics for the behavioral sciences*. McGraw-hill.

Siemens, G. and Baker, R. S. d. (2012). Learning analytics and educational data mining: towards communication and collaboration. In *Proceedings of the 2nd international conference on learning analytics and knowledge*, pages 252–254.

Sins, P. H., Savelsbergh, E. R., and van Joolingen, W. R. (2005). The difficult process of scientific modelling: An analysis of novices' reasoning during computer-based modelling. *International Journal of Science Education*, 27(14):1695–1721.

Smit, K., de Brabander, C. J., Boekaerts, M., and Martens, R. L. (2017). The self-regulation of motivation: Motivational strategies as mediator between motivational beliefs and engagement for learning. *International Journal of Educational Research*, 82:124–134.

Stone, P., Brooks, R., Brynjolfsson, E., Calo, R., Etzioni, O., Hager, G., Hirschberg, J., Kalyanakrishnan, S., Kamar, E., Kraus, S., et al. (2016). Artificial intelligence and life in 2030. *One Hundred Year Study on Artificial Intelligence: Report of the 2015-2016 Study Panel*, page 52.

Swanson, H., Irgens, G. A., Bain, C., Hall, K., Woods, P., Rogge, C., Horn, M., and Wilensky, U. (2018). Characterizing computational thinking in high school science.

The Cognition and Technology Group (1993). Anchored instruction and situated cognition revisited. *Educational Technology*, pages 52–70.

Thompson, D. B. (2006). The rational method. *David B. Thompson Civil Engineering Department Texas Tech University*, pages 1–7.

van Deursen, A., Klint, P., and Visser, J. (2000). Domain-specific languages: An annotated bibliography. *ACM Sigplan Notices*, 35(6):26–36.

van Joolingen, W. R., de Jong, T., Lazonder, A. W., Savelsbergh, E. R., and Manlove, S. (2005). Co-lab: research and development of an online learning environment for collaborative scientific discovery learning. *Computers in human behavior*, 21(4):671–688.

VanLehn, K. (2013). Model construction as a learning activity: A design space and review. *Interactive Learning Environments*, 21(4):371–413.

Vermunt, J. D. (2020). Surveys and retrospective self-reports to measure strategies and strategic processing. *Handbook of Strategies and Strategic Processing*, page 118.

Vygotsky, L. S. (1978). Mind in society: The development of higher mental process.

Wang, F. and Hannafin, M. J. (2005). Design-based research and technology-enhanced learning environments. *Educational technology research and development*, 53(4):5–23.

Weinstein, C. and Meyer, D. (1994). Learning strategies, teaching and testing. *The international encyclopedia of education*, 2:3335–3340.

Weinstein, C. E., Acee, T. W., and Jung, J. (2011). Self-regulation and learning strategies. *New directions for teaching and learning*, 2011(126):45–53.

Weinstein, C. E., Zimmermann, S. A., and Palmer, D. R. (1988). 3 - assessing learning strategies: The design and development of the lassi. In Weinstein, C. E., Goetz, E. T., and Alexander, P. A., editors, *Learning and Study Strategies*, pages 25–40. Academic Press.

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., and Wilensky, U. (2016a). Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology*, 25(1):127–147.

Weintrop, D., Holbert, N., Horn, M. S., and Wilensky, U. (2016b). Computational thinking in constructionist video games. *International Journal of Game-Based Learning (IJGBL)*, 6(1):1–17.

Wendell, K. B. and Rogers, C. (2013). Engineering design-based science, science content performance, and science attitudes in elementary school. *Journal of Engineering Education*, 102(4):513–540.

Whitelock-Wainwright, A., Laan, N., Wen, D., and Gašević, D. (2020). Exploring student information problem solving behaviour using fine-grained concept map and search tool data. *Computers & Education*, 145:103731.

Wilensky, U. (1999). Center for connected learning and computer-based modeling. In *Netlogo*. Northwestern University.

Wilensky, U. and Reisman, K. (2006). Thinking like a wolf, a sheep, or a firefly: Learning biology through constructing and testing computational theories—an embodied modeling approach. *Cognition and instruction*, 24(2):171–209.

Wilensky, U. and Resnick, M. (1999). Thinking in levels: A dynamic systems approach to making sense of the world. *Journal of Science Education and technology*, 8(1):3–19.

Wing, J. (2011). Research notebook: Computational thinking—what and why. *The Link Magazine*, 6.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3):33–35.

Winne, P. H. (1995). Inherent details in self-regulated learning. *Educational psychologist*, 30(4):173–187.

Winne, P. H. and Hadwin, A. F. (1998). Studying as self-regulated learning. *Metacognition in educational theory and practice*, 93:27–30.

Winne, P. H. and Hadwin, A. F. (2013). nstudy: Tracing and supporting self-regulated learning in the internet. In *International handbook of metacognition and learning technologies*, pages 293–308. Springer.

Winne, P. H., Jamieson-Noel, D., and Muis, K. (2002). Methodological issues and advances in researching tactics, strategies, and self-regulated learning. *Advances in motivation and achievement: New directions in measures and methods*, 12:121–155.

Wittgenstein, L. (1968). *Philosophical investigations*. Macmillan.

Wright, S. (1983). On “path analysis in genetic epidemiology: a critique”. *American journal of human genetics*, 35(4):757–768.

Wyss, V. L., Heulskamp, D., and Siebert, C. J. (2012). Increasing middle school student interest in stem careers with videos of scientists. *International Journal of Environmental and Science Education*, 7(4):501–522.

Xing, W., Pei, B., Li, S., Chen, G., and Xie, C. (2019). Using learning analytics to support students’ engineering design: the angle of prediction. *Interactive Learning Environments*, pages 1–18.

Xu, S. and Chee, Y. S. (2003). Transformation-based diagnosis of student programs for programming tutoring systems. *IEEE Trans. Software Eng.*, 29(4):360–384.

Zaki, M. J. (2001). Spade: An efficient algorithm for mining frequent sequences. *Machine learning*, 42(1-2):31–60.

Zhang, L., VanLehn, K., Girard, S., Burleson, W., Chavez-Echeagaray, M. E., Gonzalez-Sanchez, J., and Hidalgo-Pontet, Y. (2014). Evaluation of a meta-tutor for constructing models of dynamic systems. *Computers & Education*, 75:196–217.

Zhang, N. and Biswas, G. (2019). Defining and assessing students' computational thinking in a learning by modeling environment. In *Computational Thinking Education*, pages 203–221. Springer.

Zhang, N., Biswas, G., Chiu, J. L., and McElhaney, K. W. (2019). Analyzing students' design solutions in an ngss-aligned earth sciences curriculum. In Isotani, S., Millán, E., Ogan, A., Hastings, P., McLaren, B., and Luckin, R., editors, *Artificial Intelligence in Education*, pages 532–543, Cham. Springer International Publishing.

Zhang, N., Biswas, G., and Dong, Y. (2017). Characterizing students' learning behaviors using unsupervised learning methods. In *International Conference on Artificial Intelligence in Education*, pages 430–441. Springer.

Zhang, N., Biswas, G., and Hutchins, N. (n.d.). The role of strategies in student learning: Measuring and analyzing strategic learning behaviors. manuscript submitted.

Zhang, N., Biswas, G., McElhaney, K. W., Basu, S., McBride, E., and Chiu, J. L. (2020). Studying the interactions between science, engineering, and computational thinking in a learning-by-modeling environment. In Bittencourt, I. I., Cukurova, M., Muldner, K., Luckin, R., and Millán, E., editors, *Artificial Intelligence in Education*, pages 598–609, Cham. Springer International Publishing.

Zimmerman, B. J. (2000). Chapter 2 - attaining self-regulation: A social cognitive perspective. In Boekaerts, M., Pintrich, P. R., and Zeidner, M., editors, *Handbook of Self-Regulation*, pages 13–39. Academic Press.

Zimmerman, B. J. and Moylan, A. R. (2009). Self-regulation: Where metacognition and motivation intersect. In *Handbook of metacognition in education*, pages 311–328. Routledge.

Appendix A

Appendix A Summative Assessment and Rubrics

Please write your name, teacher's name, and class period if you have one.

Student Name:			
Teacher's Name:		Class Period:	

Instructions:

This activity is part of a research project.

The activity has 8 questions. Most questions have several parts.

Please be sure to read and respond to all parts of the question.

Write ***large enough*** *for another person to read your answers and use a pencil or pen that is* ***dark enough to read easily****.*

Remember…

- We have included some challenging questions on purpose.
- Don't get discouraged and try your best!

Question #	**Construct**	**Max possible points**
	Engineering	
1	Engineering problem definition and criteria	5
2	Engineering solution generation	4
3	Engineering solution comparison	4
5	Engineering fair tests	3
	Engineering sub-total	**16**
	Science	
4	Bioswale – 3D science PE	7
	Computer Science/Computational Thinking	
6	CS conditionals	4
7	Variables and conditionals	5
8	Variables, expressions and conditionals	4
	CS sub-total	**13**
	Total	**36**

1. **A grassy area downtown has been replaced by a new tennis center with cement courts. Since the center was built, a shopping area next to the center now floods after heavy rain.**

(a) Explain why the shopping area floods now.

(b) Describe one reason why this flooding is a problem for people living in the town.

(c) A city council member asks you to redesign the tennis center to solve the problem. One of the criteria for redesigning the tennis center is to keep rebuilding costs low. **List two other criteria (requirements) the city council** could give you to redesign the tennis center.

(1)	Keep rebuilding costs low
(2)	
(3)	

Q1 Rubric (5 points)

Q1a.(2 points)

Desired response: The shopping area floods because it is next to the tennis center with cement courts. Cement absorbs little rainwater causing the remaining rainwater to runoff and cause flooding.

Rubric (1a)

CODE	SCORE	DESCRIPTION
A	+1	Cement/the tennis court material absorbs little rainwater
B	+1	Surface runoff from the tennis court causes the shopping area to flood
C	0	Other incorrect
M	0	Missing

Q1b. (1 point)
Rubric (1b)

CODE	SCORE	DESCRIPTION	Example
A	+1	Student describes at least 1 resultant problem of flooding/runoff – taking away something that people may need or want (e.g., damaged cars, buildings, people getting hurt, shopping center unusable)	It is a problem because the shopping center floods and that can cause damage to the buildings and cars at the shopping center.
B	+0	Student provides a scientific explanation of how runoff occurs	The concrete does not absorb water, but the grass does
C	0	Student indicates that there is a problem, but the problem is not related to the shopping center scenario or to flooding/runoff	It is a problem because people don't have a grassy area to play People can't play tennis. Plants or crops would be destroyed.
D	0	Other incorrect	
M	0	Missing	

Q1c . (1 point per criterion = 2 points total)

Rubric (1c): Apply code separately to each criterion the student lists.

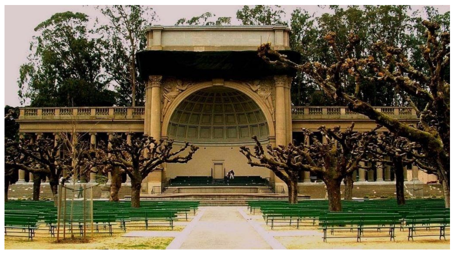
CODE	SCORE	DESCRIPTION	Example
A	+1	Student lists criterion related to design of tennis court that addresses the flooding problem	Reduce the size of the tennis complex Make the complex more grassy Reduce runoff Have a high absorption limit Add a few drains
B	+0.5	Student lists criterion related to design of tennis court, but does not address the flooding problem	Increase the size of the tennis court Paint the tennis court blue People should like the tennis center Covering for the tennis court (roof or tarp) Built it faster/get it done on time. Have a roof on top Be accessible to everyone Be wheelchair accessible Maybe use a different flooring material
C	0	Student lists criterion that is not a design criterion (such as criterion for operating the tennis center)	The tennis center should have limited hours, it should be open only when it does not rain Don’t change <something> Stop flooding

D	0	Student's response is not an example of a criterion (just a noun or too broad) Student repeats previous criterion	Surface material Color Make residents happy Relocate the court Make it better Keep the area a tennis center You can't do anything to the shopping center
M	0	Missing	

How to score?

- Do something other than grass – D
- Plant grass outside the court – A?
- Have a lot of grass around it – A?
- Only make small changes – D
- Do something other than cement – B

2. **Your town wants to build an outdoor theater at the bottom of a grassy hill. You are asked to choose ground surface materials for the surrounding seating area that will not collect water after it rains.**



Here are your criteria (requirements) for choosing the surface materials:
(1) The seating area can be asphalt, gravel, grass, or a combination of these
(2) Some water must be able to absorb (soak) into the surface
(3) The seating area must be wheelchair accessible
(4) The seating area must allow some people to sit comfortably on the ground

Properties of the surface materials

Material	Amount of water that can soak in	Wheelchair accessible?	Comfortable?
asphalt	low	yes	no
gravel	high	no	no
grass	high	no	yes

(a) What surface material or combination of materials are required to build the seating area? **You may choose more than one**.

asphalt   gravel   grass

(b) Explain how your choice of surface materials meets criteria (2), (3), and (4) above.

Q2 Rubric (4 points)

Q2a: (1 point)

Code	Score	Description
A	+1	Chooses asphalt and grass (can also choose gravel)
B	0	Does not choose asphalt OR does not choose grass
M	0	Missing

Q2b: (3 points)

Code	Score	Description	Example
A	+1	You need asphalt because it is the only material that is wheelchair accessible	The asphalt is wheelchair accessible. Asphalt meets criterion 3
B	+1	You have to have grass because it is the only material that is comfortable	The grass makes it comfortable for others to sit on the ground. Grass is nice to sit on Grass meets criterion 4

C	+1	Grass can soak up a large amount of water OR Gravel can soak up a large amount of water	The grass can absorb a large amount of water. Grass meets criterion 2 Gravel meets criterion 2
D	+0	Student picks all 3 and explains reasoning for all 3 correctly (If code D is true, Codes A,B,C will all be true)	
E	0	Other incorrect - Student does not provide a correct rationale	
M	0	Missing	

How to score?

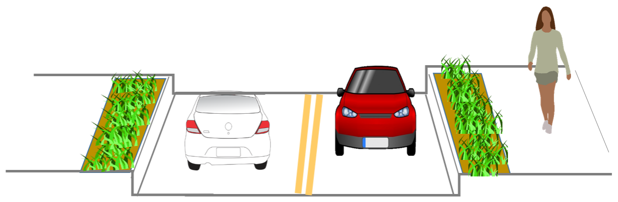
- The grass can absorb a large amount of water. The asphalt is wheelchair accessible. The grass makes it comfortable for others to sit on the ground. – A, B, C
- Asphalt is wheelchair accessible, and grass is comfortable and able to soak up water, so combining them would prove beneficial. – A, B, C
- Grass can soak in water and is nice to sit on and asphalt is wheelchair accessible. – A, B, C
- Asphalt meets criterias 2 and 3 but grass meets criteria 4. Together all criterias are met so it's perfect. – A, B
- Well asphalt has low absorption and on criteria 2 it said some water must soak in. On criteria 3 it has to be wheelchair accessible and asphalt is the only accessible. On criteria 4 it said comfortable and grass is the only comfortable one. – A, B

3. **A town plans to develop a new neighborhood by building new streets.**

The streets must meet as many of these criteria as possible:
(1) has a way to soak in rainwater
(2) has a sidewalk
(3) built at low to medium cost

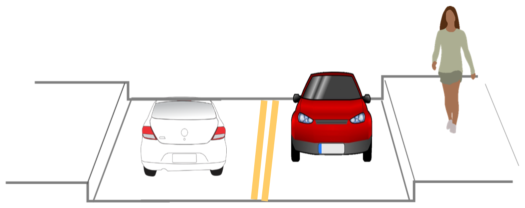
Two engineers suggest designs for the streets. Their designs are shown below.

Debbie's design



- Sidewalks
- Grassy areas beside the road
- High cost

Josephine's design



- Sidewalks beside the road
- Medium cost

(a) If the town expects **<u>a lot of rain</u>** in the future, whose solution best meets the town's needs?

Debbie's　　　　　　　　　　Josephine's

(b) Based on criteria (1), (2), and (3) above, explain why your choice is better than the other choice.

Q3 Rubric (4 points)

Rubric 3(a): (1 point)

CODE	SCORE	DESCRIPTION
A	+1	Debbie’s
B	0	Josephine’s
C	0	Both
M	0	Missing

Rubric 3(b): (3 points)

CODE	SCORE	DESCRIPTION
A	+1	Student indicates that Debbie’s design has grassy areas beside the road which can help drain the rain water
B	+1	Student indicates that Debbie's design is better because Josephine’s design does not drain water.
C	+1	Student indicates that Debbie’s design does not meet the cost criterion (or that Debbie's design preferable despite being more expensive)
D	+1	Student states that neither design is acceptable because neither meets all 3 criteria (Code D cannot be true if any of Code A, B, or C is true)
E	0	Student makes only invalid statements about either Debbie’s or Josephine’s design.
M	0	Missing

How to score?

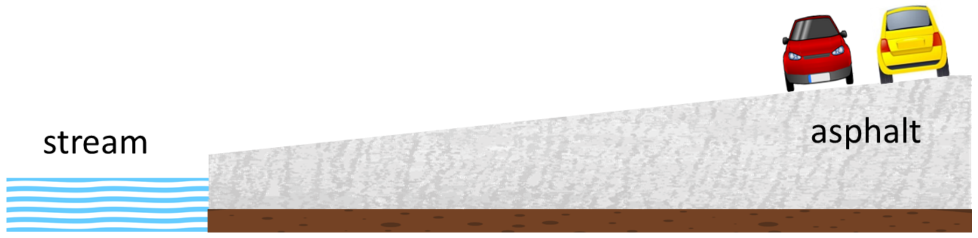
- Debbie’s design has grass to absorb the rainfall, but Josephine’s doesn’t. They both have sidewalks though. Josephine’s has medium cost and Debbie’s has high, but Josephine’s would make runoff and floods on the roads. – A,B (cost property is not tied to criterion)
- Since the town expects a lot of rain they are going to have to pay more in order to not have flooding. So I chose Debbie. - E
- It is better, because asphalt doesn’t soak in rain water, and it would be worth the cost because the number one thing is to soak in rainwater. – A, C
- The best idea based on criteria is Josephine’s but Debbie’s design is safer. If they have a lot of rain the grass would absorb the rain. - A
- If they are expecting a lot of rain they are going to need something (like grass) to absorb water so the streets don’t flood. On the downside it will cost more. – A, C
- If they were expecting rain then they could add a drainage system. - E
- Debbie’s choice is better because it meets more criteria than Josephine’s design. Her design has a way to soak in rainwater and has a sidewalk. - A
- My choice is better than the other choice because Josephine’s design wouldn’t soak up the rain, it would flood the street. - B
- The grass can soak in the water. The sidewalk has a sidewalk. It costs more but it will prevent floods. – A, C

4. A stream became more polluted after a parking lot was built next to it. Cars leave oil and other chemicals on the asphalt. Rainwater then washes these pollutants into the stream.

(a) Use **arrows and words** on the picture below to show **how the stream gets polluted**.

Your arrows should show:

- how much rainwater FALLS during a storm
- how much rainwater SOAKS INTO the surface (asphalt)
- how much rainwater FLOWS ON TOP OF the surface (asphalt)



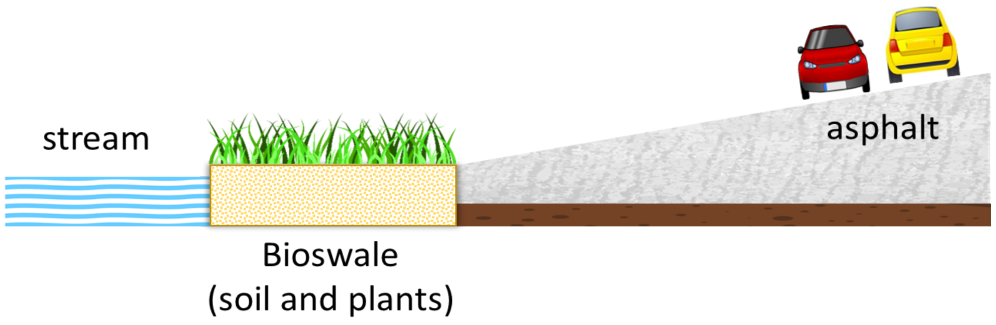
4. (continued)

To reduce the stream pollution, the town replaced some of the asphalt with a bioswale. A bioswale is an area containing soil and plants. Bioswales trap pollutants as water passes through the soil.

(b) Use **<u>arrows and words</u>** on the picture below to show **how the bioswale reduces the stream pollution**.

Your arrows should show:

- how much rainwater FALLS during a storm
- how much rainwater SOAKS INTO the surface (asphalt)
- how much rainwater FLOWS ON TOP OF the surface (asphalt)

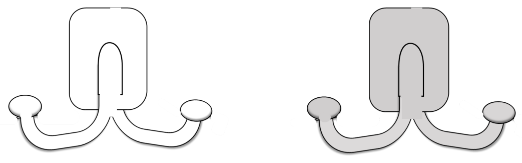


Q4 Rubric (7 points)

Holistic rubric: Same rubric applied to each of parts (a), and (b). Indications can be, but do not have to be, with arrows.

Code	Score	Feature
A	+1	Indicates rainwater falls (from the sky)
B	+1	Indicates water is absorbed into the asphalt
C	+1	Indicates water runs off asphalt
D	+0	Indicates explicitly that asphalt runoff > asphalt absorption
E	+1	Indicates water is absorbed into the bioswale
F	+1	Indicates water runs off the bioswale
G	+0	Indicates explicitly that bioswale absorption > asphalt absorption OR Asphalt runoff > bioswale runoff
H	+1	Indicates pollutants flow into the stream or on top of the asphalt
I	+1	Indicates bioswale absorbs or filters pollutants
J	0	Other incorrect or irrelevant response
M	0	Missing

5. **A school wants to mount wall hooks for students to hang their backpacks. Most backpacks are less than 40 pounds. Takumi and Hiro are testing Brand Q and Brand Z hooks to decide which are stronger.**



Brand Q **Brand Z**

Their plans to hang different weights on the hooks are below.

Takumi's plan	
Hook type	**Weight**
Brand Q	15 pounds
Brand Q	20 pounds
Brand Q	25 pounds
Brand Z	30 pounds
Brand Z	35 pounds
Brand Z	40 pounds

Hiro's plan	
Hook type	**Weight**
Brand Q	20 pounds
Brand Q	30 pounds
Brand Q	40 pounds
Brand Z	20 pounds
Brand Z	30 pounds
Brand Z	40 pounds

(a) Whose test will best help decide which type of hooks are stronger? (Circle one)

Takumi Hiro

(b) Explain your answer:

____

____

____

Q5 Rubric (3 points)

Part (a): Select ONLY one code (1 point)

Score	Code	Choice
+1	A	Hiro
0	B	Takumi
0	C	Both
0	D	Missing

Part (b): Select all that apply (Maximum: 2 points)

Score	Code	Gradescope Text	Examples
+1	A	Hiro’s plan compares Brand Q and Brand Z hooks with the same weights OR Hiro’s plan is a fair test	Hiro’s plan is better because they are comparing the same weights. Hiro’s plan is better because he can compare brand Q and brand Z for the same weights
+1	B	Takumi’s plan tests Brand Q and Brand Z hooks with different weights OR Takumi’s plan is not a fair test	Hiro’s plan is better because Takumi’s plan compares different weights.
0	C	Takumi’s plan tests weights in increasing order/smaller	Takumi because Hiro’s plan is not in order. Takumi because the weights will go in order by 5.

		increments/greater variety, or aims to identify a failure point	
0	D	Other incorrect or does not add anything to the multiple choice response	Takumi’s weights are better than in Hiro’s plan Hiro’s plan is better because he is comparing the same things. I chose Hiro because Takumi’s plan cannot determine which hooks are stronger
0	M	Missing	N/A

Sample student responses:

- Hiro’s plan is better because he is testing the same weight on each brand - A
- I think Hiro’s plan is better because if you use the same number of pounds, you can evaluate which on holds better. – A
- Because brand Z is better than Brand Q. - D
- You want them to have the same weight put on them to see which would be stronger. If you put lighter weight one brand and heavier weight on the other, then it won’t be fair. – A, B
- Takumi seemed to have more accurately weighed them. Brand Z being stronger. - D
- Because too much will not work but too little won’t help either, but going by five pound is just right (selected Takumi) - C
- Hiro’s test will explain because there is more of a variety and also Takumi’s test will not get an accurate amount. - D
- Takumi’s design shows different numbers for each brand while Hiro’s shows the same for each brand – A, B
- Hiro because her plan is more effective do the fact that all is pounds are equivalent. - A
- Takumi’s plan shows that some backpacks weigh 15 pounds so he can get that. - D
- Hiro’s plan is better because he’s testing both of the hooks with the same weight. If he didn’t, the results wouldn’t be conclusive. - A
- Hiro’s plan because it will test higher numbers very quickly - D

6. Your friend writes the following computer program:

```
If (game-score is equal to 10)
    Get the 'You're a pro' sticker
If (game-score is greater than 7)
    Get the 'Good job' sticker
```

a. Naomi has a game score of 9 points.

What sticker or stickers will she get based on the program?

i. 'You're a pro'

ii. 'Good job'

iii. Both 'You're a pro' and 'Good job'

iv. Neither

b. Janet has a game score of 10 points.

What sticker or stickers will she get based on the program?

i. 'You're a pro'

ii. 'Good job'

iii. Both 'You're a pro' and 'Good job'

iv. Neither

c. Bill has a game score of 6 points.

What sticker or stickers will he receive based on the program?

i. 'You're a pro'

ii. 'Good job'

iii. Both 'You're a pro' and 'Good job'

iv. Neither

Q6 Rubric (4 points)

Part a: (1 point)

Code	Score	Description
A	0	You’re a pro
B	+1	Good job
C	0	Both ‘You’re a pro’ and ‘Good job’
D	0	Neither
E	0	Multiple choices selected
M	0	Missing

Part b: (2 points)

Code	Score	Description
A	+1	Only ‘You’re a pro’
B	+1	Only ‘Good job’
C	+2	Both ‘You’re a pro’ and ‘Good job’
D	0	Neither
M	0	Missing

Part c: (1 point)

Code	Score	Description
A	0	You’re a pro
B	0	Good job
C	0	Both ‘You’re a pro’ and ‘Good job’
D	+1	Neither

E	0	Multiple choices selected
M	0	Missing

7. **An amusement park ride needs riders to be at least 46 inches tall.**
   The program below uses the variable Height to determine whether the person can ride.

   Use the variable **Height** to fill in the IF statements to complete the program.

   IF

   Say “You are not tall enough to ride”

   IF

   Say

   “Congratulations, you are tall enough to ride”

Q7 Rubric (5 points)

Rubric for first IF (), Say, “You are not tall enough to ride”: (2 points)

Code	Score	Description	Example
A	+2	Correct comparison to 46 (less than) expressed including use of variable “Height”	Height < 46 Height less than 46 Height < 46 inches Height <= 45
B	+1	Correct comparison to 46 (less than) expressed in English language without using variable “Height”	Rider’s height is less than 46 inches You are less than 46 Under 46 inches
C	+1	Uses the variable “Height” with incorrect comparison to 46	Height is 46 inches or less Height > 50

D	0	Mention a specific height less than 46 inches	42 inches
E	0	Other incorrect	
M	0	Missing	

Rubric for second IF (), Say, “Congratulations, you are tall enough to ride”: (3 points)

Code	Score	Description	Example
A	+3	Correct comparison to 46 (greater than equal to) expressed including use of variable “Height”	Height >= 46 Height greater than or equal to 46 Height >= 46 inches Height = 46 inches or height > 46 inches
B	+2	Correct comparison to 46 (greater than equal to) expressed in English language without using variable “Height”	Rider’s height is greater than or equal to 46 inches You are 46 or more 46 and above
C	+2	Includes variable “Height” but gets only 1 part of the condition, either greater than 46, or equal to 46	Height > 46 Height = 46 Height greater than 46 Height > 46 inches

D	+1	Gets only 1 part of the condition, either greater than 46, or equal to 46, and does not use the variable “Height”	Rider’s height is greater than 46 inches You are 46 Above 46
E	+1	Uses the variable “Height” with completely incorrect comparison to 46	Height > 52 Height < 46
F	0	Mention a specific height equal to or greater than 46 inches	46 inches 50 inches
G	0	Other incorrect	
M	0	Missing	

Sample student responses:

- Height < 46
  - Height > 46
- The program uses Height
  - The program uses Height
- You are not tall enough to ride
  - Congratulations, you are tall enough to ride
- 42 inches
  - 49 inches
- Height < 46 inches
  - Height = 46 inches or height > 46 inches
- (height lower than 46)
  - (height is 46 inches or above)
- You are 46 or less
  - You are 46 or above
- 42
  - 46

- You are 45 inches or less
    - You are equal to 46 in or greater
- Under 46 inches
    - 46 inches or taller

8. Jenna is saving money to buy her friend a gift that costs $20.

The variable **MoneySaved** is how much money Jenna has already saved.

The variable **MoneyNeeded** is how much money Jenna still needs.

For example:

- If MoneySaved is $15, MoneyNeeded is $5
- If MoneySaved is $25, MoneyNeeded is $0

Use the variables **MoneySaved** and **MoneyNeeded** to complete the program below so that it calculates how much more money is needed to buy the gift.

If (**MoneySaved** is less than 20)

If

(**MoneySaved** is equal to 20)

If (**MoneySaved** is greater than 20)

Q8

Rubric (4 points)

Rubric for first IF (MoneySaved is less than 20): (2 points)

Code	Score	Description	Example
A	+2	General solution using correct variables and expressions	MoneyNeeded = 20 - MoneySaved
B	+1	Correct expression but only one of the variables expressed correctly	20 – MoneySaved MoneyNeeded = 20 - Savings
C	+1	Uses correct variables, chooses a specific value for MoneySaved and correctly calculates value of MoneyNeeded for that case	If MoneySaved $10, MoneyNeeded is $10 MoneySaved is 5$ then need $15
D	0	Follows numbers from example provided in prompt	MoneyNeeded is $5
E	0	Other incorrect	Get more money
M	0	Missing	

Rubric for second IF (MoneySaved is equal to 20): (1 point) Make it 2 points, one for the variable and one for the constant (value or string) or expression

Code	Score	Description	Example

A	+1	General solution using correct variable(s)	MoneyNeeded = 20 – MoneySaved
			MoneyNeeded = 0

B	+0.5	General solution using 1 incorrect variable name	20 – MoneySaved

C	+0.5	Chooses a specific value for MoneySaved and correctly calculates value of MoneyNeeded for that case	If MoneySaved is $20, MoneyNeeded is $0
D	0	Other incorrect	Just enough money $0
M	0	Missing	

Rubric for third IF (MoneySaved is greater than 20): (1 point)

Code	Score	Description	Example
A	+1	General solution using correct variable	MoneyNeeded = 0
B	+0.5	Chooses a specific value for MoneySaved and correctly calculates value of MoneyNeeded for that case	If MoneySaved is $30, MoneyNeeded is $0
C	0	Other incorrect	No more money needed $0
M	0	Missing	

How to score:

MoneyNeeded = 20 – MoneySaved (3 points)
20-Moneysaved: 2 points
20-Savings: 1 point
If MoneySaved $10, MoneyNeeded is $10: 1 point

MoneySaved is 5$ then need $15: 1 point

Sample student responses:

- MoneyNeeded = 20 – MoneySaved
    - MoneyNeeded = 0
    - MoneyNeeded = 0
- If MoneySaved $10, MoneyNeeded is $10
    - If MoneySaved is $20, MoneyNeeded is $0
    - If MoneySaved $30, MoneyNeeded $0
- Money needed is $5.00
    - Money needed is $0.00
    - Money saved is $0.00
- MoneyNeeded
    - MoneySaved
    - MoneySaved
- MoneySaved is 5$ then need $15
    - Money 25$ then 0$ needed
    - Money 25$ then 0$ needed
- MoneyNeeded is = $20 or <$20
    - MoneyNeeded is = $0
    - MoneyNeeded is = $0
- Subtract 25 from number
    - MoneyNeeded is $5
    - Puchase
- 5$ needed
    - $0 needed
    - $0 needed
- MoneyNeeded
    - MoneySaved
    - MoneySaved
- Needed is 20
    - Needed 0
    - Needed 0
- Get more money
    - You are just right
    - You have money left over