International Journal of Artificial Intelligence in Education (2022) 32:931–970
https://doi.org/10.1007/s40593-021-00275-x

ARTICLE

Measuring and Analyzing Students' Strategic Learning Behaviors in Open-Ended Learning Environments

Ningyu Zhang[1] · **Gautam Biswas**[1] · **Nicole Hutchins**[1]

Accepted: 4 August 2021 / Published online: 24 September 2021

Abstract

Strategies are an important component of self-regulated learning frameworks. However, the characterization of strategies in these frameworks is often incomplete: (1) they lack an operational definition of strategies; (2) there is limited understanding of how students develop and apply strategies; and (3) there is a dearth of systematic and generalizable approaches to measure and evaluate strategies when students' work in open-ended learning environments (OELEs). This paper develops systematic methods for detecting, interpreting, and analyzing students' use of strategies in OELEs, and demonstrates how students' strategies evolve across tasks. We apply this framework in the context of tasks that students perform as they learn science topics by building conceptual and computational models in an OELE. Data from a classroom study, where sixth-grade students ($N = 52$) worked on science model-building activities in our Computational Thinking using Simulation and Modeling (CTSiM) environment demonstrates how we interpret students' strategy use, and how strategy use relates to their learning performance. We also demonstrate how students' strategies evolve as they work on multiple model-building tasks. The results demonstrate the effectiveness of our strategy framework in analyzing students' behaviors and performance in CTSiM.

Keywords Learning strategies · Simulations · Open-ended learning environments · Learning by modeling

✉ Gautam Biswas
gautam.biswas@vanderbilt.edu

Ningyu Zhang
ningyu.zhang@vanderbilt.edu

Nicole Hutchins
nicole.m.hutchins@vanderbilt.edu

[1] Department of EECS, Institute for Software Integrated Systems, Vanderbilt University, Nashville, TN, USA

Introduction

The term *strategy* comes from ancient Greek and refers to plans for winning a war. In the context of educational research, strategies generally represent systematic plans or sub-plans students develop for achieving goals (Oxford, 2011; Vermunt, 2020). In their simplest form, strategies are *conditional* constructs that can be represented as *if-then-else* rules (Winne et al., 2002). Strategies have been receiving attention in the teaching-and-learning literature since the mid-1980s as metacognitive processes (Alexander et al., 1998; Flavell, 1979; Nisbet & Shucksmith, 2017) that foster successful learning (Weinstein et al., 2011; Panadero & Alonso Tapia, 2014).

More broadly, the research literature in educational psychology and the learning sciences discuss different types of strategies, such as (1) cognitive strategies (Donker et al., 2014; Pressley et al., 1989); (2) metacognitive strategies (Schraw et al., 2006); (3) affect regulation strategies (Parkinson & Totterdell, 1999), and (4) management strategies (Donker et al., 2014) The literature also discusses methods used for instruction on strategies (Donker et al., 2014; Weinstein et al., 2011), and evaluating the effectiveness of strategies (Garner, 1988) in multiple subject domains that include the language arts (Oxford, 2011), writing (Roscoe et al., 2019), mathematics (Donker et al., 2014; Liu et al., 2016), science (Alesandrini, 1981; Tsai & Tsai, 2003), and massive open online courses (MOOCs) (Matcha et al., 2019).

Despite its broad applications to learning, it is hard to glean an operational definition of strategies and methods for evaluating strategies from existing literature, especially with respect to their application in computer-based learning environments (Wang & Hannafin, 2005). We will address these issues by developing a framework for modeling, measuring, and assessing learners' strategies from their learning and problem-solving activity data collected in log files, while students work on building conceptual and computational models of scientific processes in an open-ended learning environment (OELEs) (Basu et al., 2014; Roscoe et al., 2013). This paper makes two primary contributions:

1. Developing an operational definition of learning strategies in the context of students' activities in OELEs, and more specifically *learning-by-modeling* environments, where students learn their science content by building conceptual and computational models of scientific processes. "A Strategy Framework for the CTSiM OELE" presents our precise definition of strategies and some key considerations for analyzing learning strategies in our OELE environments. Our definition is compatible with well-established strategy frameworks reported in the literature; and
2. Extending and applying our previous work on (1) an exploratory task-based methodology to infer and analyze students' learning strategies from their activity sequences when students worked on causal modeling tasks (Kinnebrew et al., 2017), and (2) an adaptive scaffolding approach to help students develop and apply specific strategies as they worked on computational modeling tasks (Basu et al., 2017) in OELEs. We discuss this in greater detail in "Problem Statement".

We apply the framework to detect and analyze students' learning strategies when they work on conceptual and computational model-building and problem-solving tasks in ***C**omputational **T**hinking using **Si**mulation and **M**odeling* (CTSiM), a *learning-by-modeling* environment that fosters synergistic learning of science and computational thinking (CT; (Sengupta et al., 2013)). *Learning-by-modeling* environments represent a specific class of OELEs that provide learners with rich opportunities to develop and practice using strategies. We describe the components of the CTSiM system and explain how students go about building their computational models in this environment. We then discuss the student activity data collected in the form of system logs, and the analyses we conduct with the data to answer three primary research questions:

- **RQ 1:** *Can we detect and characterize students' use of learning strategies in the CTSiM OELE?*
- **RQ 2** *Can we track how students' use of learning strategies evolve as they build computational models for different scientific processes?*
- **RQ 3** *How do we evaluate the effectiveness of strategies that students employ when working with CTSiM?*

The rest of the paper is organized as follows. "Background and Related Work" reviews learning strategies discussed in the literature, summarizes the role of strategies in self-regulated learning (SRL) frameworks and illustrates strategy use in some representative OELEs. The section also presents our problem statement and outlines how we extend past work on strategy detection and analysis. "Learning by Modeling: The CTSiM System" discusses learning-by-modeling environments and describes the CTSiM system as a computational modeling environment. "A Strategy Framework for the CTSiM OELE" synthesizes our definition of cognitive and metacognitive strategies and presents systematic approaches for detecting and interpreting students' strategy use during model-building and problem-solving tasks in CTSiM. We also outline the methods and measures we have developed for analyzing students' learning behaviors and their performance using formative and summative assessments. "Results" presents the results of an experimental study conducted with the CTSiM system in a middle school classroom. We analyze the data collected from the study to answer the three **RQ**'s stated above. Finally, "Discussion and Conclusions" presents our conclusions from the study and discusses the generalizability of our framework and the directions for future work.

Background and Related Work

Strategies are ubiquitous and essential for effective learning at all levels. The cognitive and learning sciences literature describes strategies as an essential component of critical thinking and learning (Boekaerts, 1996; Zimmerman, 2000). Better-performing students are known to be more strategic in their approach to learning than less competent learners (Derry, 1990), and the use of strategies helps learners

become more self-reliant and fosters life-long learning (Cornford, 2002; Weinstein & Meyer, 1994). Furthermore, explicit instruction of cognitive and metacognitive strategies improves overall learning performance (Weinstein & Meyer, 1994; Zhang et al., 2014). In this section, we briefly review two related topics: (1) the scope of learning strategies and strategy frameworks, and (2) strategy use in OELEs. "Problem Statement" presents our problem statement and how our approach extends past work on strategy detection and analysis.

Strategies and Strategy Frameworks

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). They are known to positively influence information processing and the development of new skills in support of learning (Mayer, 1988; Pressley et al., 1989; Oxford, 2011; Obergriesser & Stoeger, 2020). 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 (Weinstein et al., 2011; Winne & 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), "*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. Both constructs are associated with orchestrating cognitive resources and skills, but they differ in their generality and purpose (Weinstein & Meyer, 1994; Cornford, 2002). Typically, cognitive strategies are goal-directed, intentionally invoked, situation-specific, and not universally applicable (Weinstein & Meyer, 1994). On the other hand, metacognitive strategies involve more generally applicable processes, such as planning, monitoring, and evaluating (Donker et al., 2014; Cornford, 2002). Operationally, cognitive strategies include the knowledge of 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 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 (p. 24 Nisbet & Shucksmith 2017). More generally, metacognition can be described as deliberating on the use of particular cognitive processes and how to combine them to accomplish larger tasks (Winne, 1995).

Metacognitive monitoring serves as the bridge connecting cognition and metacognition, 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 focus on cognitive and metacognitive strategies.

The study of strategies is often contextualized within the broader framework of *self-regulated learning* (SRL) (Schunk & 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).

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. 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). 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 (Panadero & Alonso Tapia, 2014; Zimmerman, 2000).

The COPES model (conditions, operations, products, evaluations, and standards) of Winne and Hadwin (1998) 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. It 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 & Hadwin, 1998; Winne et al., 2002). In the COPES model, a strategy is defined as a collection of *if-then* rules (each individual rule is a tactic) that create larger patterns and extend to *if-then-else* rules over time (Winne et al., 2002). Therefore, strategies are more complex than tactics in their structure, and they also have a larger scope that yields more information that can be used for feedback during learning (Winne et al., 2002).

In the recursive loop defined by 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 strategies used, and then tunes or restructures some of the strategies to make them more effective for the goals and plans of the learning task (Winne et al., 2002).

Strategies and Open-Ended Learning Environments (OELEs)

OELEs are a class of computer-based learning environments that adopt a constructivist epistemology to support the acquisition of knowledge and skills (Land, 2000).

They provide learners with opportunities to practice problem-solving skills in real-world contexts (Wang & Hannafin, 2005). A learning environment is *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).

The exploration of learners' strategy use in OELEs is linked to understanding the "*dynamic and cyclical nature of self-regulation*" processes they employ (Schmeck, 2013, p. 5). Wittgenstein stated that "An 'inner process' stands in need of outward criteria" (Section 530 ; Wittgenstein, 1968). 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 strategy use. These methods, based on students' activity sequences, enrich the measurements of strategy use using self-reports, think-aloud utterances, and interviews (Schmeck, 2013; Vermunt, 2020; Whitelock-Wainwright et al., 2020).

Deployment of OELEs has made it possible to collect detailed traces of students' interactions within the learning environment and linking them to outcomes (Schmeck, 2013). A series of methods have been designed to infer students' use of strategies by distilling contextualized and operationalized information from the traces recorded in log files (Azevedo et al., 2013; Bannert et al., 2014; Hadwin et al., 2007; Kinnebrew et al., 2017; Segedy et al., 2015b). These analytical frameworks and methods lay the foundation for tracking and evaluating how students' use of strategies evolves with time. As a result, OELEs have 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 & du Boulay, 2016), nStudy (Winne & Hadwin, 2013), MetaTutor (Azevedo et al., 2010), and Betty's Brain (Biswas et al., 2016).

OELEs make provisions for learners to choose their own learning goals and/or the methods to achieve their goals (Hannafin et al., 1999; Hannafin et al., 2014). For example, in MetaTutor, students must strategically select the focal concepts to read and comprehend due to the limited amount of time and the complexity of the tasks. In the nStudy environment, students learn and comprehend digital inscriptions (de Vries et al., 2009) in a web application. Students' learning goals and the means they employ to accomplish these learning goals are not predetermined. In the rest of this subsection, we review the four OELEs with an emphasis on how they support students' developing and applying strategies as they learn by performing tasks in these environments.

The **Ecolab** family of constructivist learning environments (e.g., Ecolab, M-Ecolab, and Ecolab II) focus on middle school's science topics, such as food chains and food webs (Luckin & du Boulay, 2016). Learners using Ecolab can select 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 & du Boulay, 2016). Ecolab presents learning tasks to the students, utilizing

the learner's Zone of Proximal Development (ZPD) (Vygotsky, 1978) to facilitate the learning. The environment supports the development of metacognitive and goal-orientation strategy constructs, such as help-seeking and self-monitoring.

A variation of the environment, M-Ecolab, adapts to learners' motivational state to select the form of metacognitive strategy scaffolding to provide to them (Luckin & du Boulay, 2016; Mendez et al., 2005; Luckin & Hammerton, 2002). For example, help-seeking, an indispensable metacognitive strategy, helps students develop mastery and comprehension skills (Newman, 1990). Studies with Ecolab II have shown that this strategy supports students learning of domain content, and helps them to improve their task selection and self-monitoring processes. This is especially true for low-ability students who may lack prior metacognitive skills (Luckin & Hammerton, 2002; Harris et al., 2009).

nStudy, a web-based application, provides learners with opportunities to practice important learning skills and SRL processes while studying with digital content in the form of multimedia web pages, PDF files, and video clips (Winne & Hadwin, 2013). Learners are provided with tools for highlighting and annotation to create learning artifacts, such as bookmarks, notes, and concept maps (de Vries et al., 2009). The system helps students develop learning strategies related to setting goals, seeking information, and creating, modifying, and deleting learning objects (Beaudoin & Winne, 2009).

In nStudy, a cognitive strategy (e.g., note-taking, writing, and test preparation and review) is represented as a learning object that contains the following information: (1) why the strategy helps, (2) when and how to use it, and (3) examples of its use (Beaudoin & Winne, 2009). Learners can define their learning strategies and link them to their own learning artifacts (e.g., bookmarks and notes). nStudy also provides learners with opportunities to self-evaluate the effectiveness of a strategy or to create criteria for evaluating a strategy (Beaudoin & Winne, 2009). Students' reading and annotating behaviors in different learning phases (e.g., task understanding, planning, monitoring, evaluating) are captured as authentic learning traces (Hadwin et al., 2007). These collected traces reflect the students' cognitive and metacognitive events in the learning environment (Hadwin et al., 2007).

The **MetaTutor** OELE promotes the use of strategies to foster students' learning of complex topics in biology, for example, the human circulatory system (Azevedo et al., 2010; Taub et al., 2018). The system provides students with a large number of content pages from which they can collect information to write a summary on the topic of study. Learners have a limited amount of time for reading the complex content and extracting the relevant material to generate the summary on the specified topic. Therefore, they need to be strategic and selective in their reading and other learning activities in MetaTutor.

In addition to the hypermedia resources, the system includes a team of four virtual pedagogical agents who help students develop cognitive and metacognitive strategies: (1) Gavin the Guide, (2) Pam the Planner, (3) Mary the Monitor, and (4) Sam the Strategizer. Students can query the four agents at any time. Depending on their area of expertise, particular agents then provide scaffolding and feedback to the learners

on developing and using strategies. For example, Pam the Planner helps students activate prior knowledge and manage their learning goals, while Mary the Monitor may remind students to invoke key strategies linked to evaluating their understanding of the content (Azevedo et al., 2010; Taub et al., 2018).

MetaTutor assesses students' strategy use and provides relevant feedback. For example, students may indicate that they would like to know how to activate prior knowledge or to evaluate their learning of the biology content (Taub et al., 2019). MetaTutor uses the task the student has explicated as the context for providing feedback. MetaTutor also links students' learning to their emotions and affect. For example, Taub et al. (2019) used correlations to establish links between students' positive and negative emotions along with their note-taking and summarizing activities. Correlation analyses conducted in a series of experimental studies using MetaTutor (Cloude et al., 2018; Greene & Azevedo, 2010; Taub et al., 2018; Taub et al., 2019) have shown that students' learning gains are linked to their strategy use. For example, one study found that college students' metacognitive awareness increased as they progressed in their learning tasks (Greene & Azevedo, 2010).

The **Betty's Brain** learning environment (Biswas et al., 2016; Leelawong & Biswas, 2008) is an OELE that helps students acquire knowledge and understanding of scientific phenomena, such as climate change and the human body thermoregulation, by constructing causal models. The system adopts a *learning-by-teaching* paradigm (Biswas et al., 2005), where students construct the causal model to teach a virtual teachable agent, generically named Betty. A mentor agent monitors students' work, and when needed provides help in the form of cognitive and metacognitive learning strategies (Munshi et al., 2018). As she is being taught a particular topic, for example, the causes and effects of climate change, 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 require a number of strategies that students can employ to find and correct errors in the model. Studies demonstrate that students learning (measured as pre- to post-test learning gains) 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.

As an OELE, Betty's Brain provides students with rich opportunities to practice their metacognitive strategies for solving complex and open-ended problems (Segedy et al., 2015b). A series of middle school classroom studies with Betty's Brain have demonstrated that students achieved significant pre- to post-test learning gains on science content (Leelawong & 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) used more effective learning strategies (Kinnebrew et al., 2014; 2017).

Problem Statement

The literature has established that learning strategies are an integral part of SRL frameworks. However, despite their importance and ubiquity, research on strategies needs to address:

1. The *Grounding problem*. Learning strategies are generically defined as *skills or methods to enhance performance* but they are not grounded in operational forms. As a result, they cannot be measured on a consistent, numerical scale;
2. The *Generalizability problem*. Learning strategies have been studied in specific and disjoint learning domains, such as reading, writing, and mathematics, but a general framework for representing strategies has not emerged.
3. The *Measurement problem*. There is a dearth of systematic approaches for measuring and evaluating strategy use from learners' activity data in computer-based learning environments and in particular, OELEs. Most related work has been directed toward analyzing undergraduate students' strategy use in flipped classrooms or MOOC settings.

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 & Greene, 2018; Fincham et al., 2018). 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); (2) the interviewer interprets and represents the learner's response through their own heuristics lenses (Winne et al., 2002); and (3) learners report their *perceived* learning methods rather than their actual learning processes (Fincham et al., 2018). Therefore, the observation of a phenomenon (strategy definition and use) is sensitive to and can be greatly impacted by the data collection method (Dent, 1999).

In this work, we directly address two of the three issues listed above, i.e., the grounding problem and the measurement problem. To elaborate further, one of our goals in this paper is to extend and generalize our previous work on detecting and interpreting students' learning strategies in the Betty's Brain environment (Kinnebrew et al., 2017; Segedy et al., 2015a). The model construction process in Betty's Brain requires students to build a causal map. In the work reported in this paper, students have to synergistically combine their knowledge of science and CT concepts and practices to build, test, debug, and analyze computational models of scientific processes. These synergistic processes require the use of more involved and comprehensive strategies (Hutchins et al., 2020a; Sengupta et al., 2013; Snyder et al., 2019).

In this paper, we develop a task-oriented strategy framework for the CTSiM learning-by-modeling environment using cognitive task analysis (Schraagen et al., 2000). Our exploratory, data-driven approach combines coherence analysis and sequence mining methods to identify and interpret effective and ineffective strategies as students work on open-ended learning tasks. Our approach also allows us to track changes in students' strategy application over time (i.e., across a sequence of tasks).

We define open-ended learning tasks as those in which students have a choice in pursuing their learning and problem-solving tasks. In general, learners can leverage

resources provided in the environment to acquire, understand, and apply the knowledge needed to solve or complete a problem or goal. They are also provided with resources that help them test and check on their evolving solutions to assigned problems. As discussed in the literature review ("Background and Related Work"), this includes inquiry environments (e.g., Ecolab), study tools (e.g., nStudy, MetaTutor), and constraint-based models (e.g., Betty's Brain), and their accompanying resources (e.g., digital content, pedagogical agents, expert models). Further, this approach has been extended to studying students' open-ended learning and problem-solving strategies in game-based environments, such as Crystal Island (Taub et al., 2017) and Tuglet (Käser & Schwartz, 2020).

We present our task model in "The Task and Strategy Models of CTSiM" that maps student actions in our OELE to three primary cognitive processes (Information Acquisition, Solution Construction, and Solution Assessment). Open-ended learning tasks that expect students to employ similar cognitive processes can leverage the presented framework. Furthermore, learning environments that log students' activities in the context of the tasks they are performing can employ the methods we describe below for analyzing students' learning behaviors and strategies.

Along these lines, our framework generalizes the previous work presented in Basu et al (2017), where we studied students' use of specific, pre-identified strategies and developed adaptive scaffolding methods to help them learn these strategies effectively. As a result, the exploratory analysis presented for discovering students' strategies and linking them to their performance tackles the *grounding problem* through the systematic analysis of activity data during model construction and the *measurement problem* by leveraging that analysis in the context of student knowledge construction and learning during open-ended problem solving in different domains. Therefore, this presents an example of generalization from our previous framework, supporting the *generalizability problem* based on the framework presented in the context of OELEs in the previous paragraph.

Learning by Modeling: The CTSiM System

Computational modeling and the use of computational thinking (CT) concepts and practices (Wing, 2006; Chao, 2016) present an effective way to promote an understanding of dynamic systems (Wilensky & Reisman, 2006). For example, Weintrop et al. (2016) have discussed the CT practices associated with understanding the relationship within a system and investigating systems at different levels to foster students' development of systems thinking in STEM classrooms.

The deep understanding of systems with emergent behaviors, along with the proficient application of learning strategies, is essential to prepare students for future scientific endeavors. Science education standards have 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). However, students often face difficulties in performing system-level analyses (Basu et al., 2015; Basu et al., 2016; Cheng et al., 2010; Chi, 2005; Chi et al., 2012; Wilensky & Resnick, 1999).

Some of the difficulties can be attributed to the confusion in the relationship between components, e.g., the realization that emergent behaviors arise as higher-level patterns derived from aggregated interactions of lower-level components (Jacobson & Wilensky, 2006). Other forms of misunderstanding, e.g., understanding the notion of equilibrium can be attributed to commonsense conceptions of systems, i.e., for a system to be in equilibrium it must be at rest. Novice learners are prone to assigning 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). Roscoe et al. (2013) summarized and analyzed novice learners' development of *shallow* strategies, especially when SRL support was not offered by the OELE. Overall, the challenges that novice learners face are multi-fold, consisting of content-related difficulties and the lack of efficient and effective learning strategies to overcome difficulties and incorrect conceptions (VanLehn, 2013).

Learning Science by Building Computational Models

Educators and cognitive scientists have suggested that the exercise of building computational models may help students gain a deep understanding of scientific phenomena by overcoming common difficulties and misunderstandings (NRC, 2010; Wilensky & Reisman, 2006). Modeling is an essential practice in STEM domains that supports conceptualization, understanding, and reasoning about complex scientific phenomena. 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; VanLehn, 2013). Model-based, constructivist curricula are more effective when compared with traditional approaches (NRC, 2000).

The CTSiM OELE

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 by building computational models (Sengupta et al., 2013; Zhang et al., 2017; Zhang & Biswas, 2018). Students perform five primary learning-oriented tasks as they build computational models of science phenomena: (1) acquiring information of the domain and CT-related content by reading the built-in hypertext resource library; (2) building conceptual models of the system by defining the components (e.g., agents and the environment) and their properties and behaviors using an agent-based modeling framework; (3) constructing executable computational models that define the agents' interactions with the environment and with each other using a block-based modeling language; (4) running simulations with the computational models to analyze and debug the agents' behaviors; and (5) comparing the behaviors of their computational models to the behaviors generated by an expert reference model (Basu et al., 2017).

Figure 1 illustrates the interfaces for five of the primary activities that students perform in the system: (1) Acquiring information from the system library (top-left); (2) Building a conceptual model by identifying components of the system (e.g., the molecules and the container) and selecting their properties (e.g., the position, direction of travel, and velocity) (top-right) from pre-populated drop-down lists of the agent or environmental properties; (3) Dragging-and-dropping computational blocks to implement the *update-speed-and-direction* behavior of the dye molecule agents in the diffusion process (center-left); (4) Testing the computational model by running simulations and observing the generated model behaviors (center-right); and (5) Comparing the model behaviors to those of an expert model running synchronously to deduce if the student's model behaviors differ from that of the expert model (bottom).[1] In the *testing* and the *comparison* interface, the students can set up the simulation with various combinations of parameters to ensure that their model implementation covers enough edge-cases and that the simulation invariant is held true in all cases. In this particular example (bottom row of Fig. 1), the student's model terminated the simulation when a local equilibrium on the concentration gradient was met, which was identified as a common misconception among novice learners by Chi et al. (2012). The plots in the simulation interface are presented in a dashboard format, and students can check the simulation status, and also look for potential modeling errors (e.g., the average speed and total energy of the particles should remain relatively constant).

In addition, it is important to note that students have access to all interfaces at all times, supporting the open-ended nature of the approach. This means that students can run comparisons with an expert model, regardless of whether they have completed any steps in the conceptual and computational modeling interfaces. More specifically, if the student has not created a model when they run a comparison, the expert model executes with a default set of parameters. If the student has created a partial model, e.g., they have specified some initial parameters but specified no behaviors, then the expert model runs using the student-specified parameter values as an initialization. The initialization also impacts the view of the student simulation, but the agents in the student model show no dynamic behaviors, because the student has not specified any. If the student specifies some or all required behaviors (correctly or incorrectly) both models are executed in lock-step to the extent it is possible until one terminates, at which point both models are stopped. As a result, the student gets to compare their model behaviors against the expert model behaviors in animation and through the generated plots. However, the student never sees the code for the expert model.

While performing the conceptual and computational model-building activities, students learn the domain content along with CT concepts (e.g., variables, loop structures, and conditionals) and practices (e.g., systems thinking, problem decomposition, and debugging) (Basu et al., 2016; Zhang et al., 2017; Weintrop et al., 2016). Much like other OELEs, CTSiM offers learners the freedom to decide their own learning goals and learning trajectories that demonstrate their choice of information

[1] The students are not able to see the code for the expert model.

Fig. 1 The graphical user interface and learning activities of CTSiM. The top pane illustrates the Resources window; The middle pane illustrates the Solution Construction (model-building) window; and the bottom pane shows the Solution Assessment (study and compare model behaviors) window

acquisition, model-building, and model-checking tasks. These choices depend on students' metacognitive monitoring and reflect students' understanding of the use of various learning strategies.

We have developed science learning units in the CTSiM system for middle school classrooms. These include kinematics (relations between acceleration, velocity, and position of objects) and dynamics (introduction of force and momentum) units in mechanics, diffusion processes (movement of suspended particles in a liquid), and ecology (fish tank system). In the diffusion unit, students model the processes of (1) random motion of individual particles, (2) particle collisions with one another, and (3) particle collisions with the container wall. We believe that constructing correct computational models of the component processes of colliding particles and studying the resultant aggregated behaviors helps students understand the emergent behaviors of diffusion processes. It also helps them overcome the misunderstandings that may be linked to (1) false intentionality in a system (Chi et al., 2012) and (2) centralized control (Cheng et al., 2010).

A Strategy Framework for the CTSiM OELE

In this section, we introduce our methods for modeling, measuring, and evaluating students' strategies in OELEs. 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 from 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**.

As an open-ended and choice-rich environment, CTSiM provides students with a multitude of opportunities to develop and practice their learning strategies for building models of scientific processes. In this work, we focus on **cognitive strategies** (e.g., acquiring and organizing information into computational structures) and **metacognitive strategies** (e.g., monitoring and executing control, i.e., selecting, modifying, and replacing cognitive strategies). The use of strategies is latent, thus, they cannot be observed or measured directly. However, the use of strategies is reflected in the sequence of actions students perform in the environment, and they indirectly manifest students' declarative, procedural, and conditional knowledge (Schraw et al., 2006).

In previous work, we have applied cluster analysis methods to categorize the different learning behaviors students exhibit in CTSiM and found that certain behaviors lead to better model-building performance and better pre- to post-test learning gains in the domain and CT concepts (Zhang et al., 2017). We have developed similar approaches to characterizing students' aggregate behaviors in the Betty's Brain environment (Gauch & Biswas, 2016; Segedy et al., 2015a; 2015b) and have obtained similar results. In our analyses, we found that the aggregated behavior descriptions (e.g., the time spent on certain learning activities) can mask the actual strategies students use because of the loss of contextual information in the aggregated actions (Kinnebrew et al., 2013).

The approach presented in this paper defines strategies and targets their **systematic evaluation as conscious and controllable sequences of actions students perform to facilitate and enhance their task performance**. Therefore, the approach identifies common action sequences that students employ to accomplish their tasks and enhance their overall model-building performance (in this work, their goal is to build models whose behaviors closely match expert-generated solutions).

To do so, we adopt *Coherence Analysis* (CA) (Segedy et al., 2015b) along with a combined theory- and data-driven framework for identifying, interpreting, and analyzing students' common sequences of actions in CTSiM and for evaluating the effectiveness of those sequences (e.g., strategies) in supporting problem solving. This approach generalizes and extends an approach that was previously developed and applied to the Betty's Brain system (Kinnebrew et al., 2017; Segedy et al., 2015a). We detail the steps of this approach in the following subsections.

The Task and Strategy Models of CTSiM

The first step in this process is to generate a hierarchical task model that creates a mapping between students' actions in CTSiM to the three primary sub-tasks (cognitive processes) they employ for building and analyzing computational models: (1) Information Acquisition (IA), (2) Solution Construction (SC), and (3) Solution Assessment (SA). Figure 2 (adapted from Basu et al. 2017) shows the task model with the cognitive processes expressed at different levels of detail. Students' logged, time-stamped actions map directly to the leaf nodes of the hierarchy. There are no horizontal links in the task model, implying that students are free to perform their activities in any order they wish to accomplish their tasks.

Students perform IA actions that include *Read Page* and *Search Information*, using the Resources interface (the top-left of Fig. 1). They perform SC actions that include *conceptual* (top-right, Fig. 1) and *computational* (middle-left, Fig. 1) model construction and model editing actions. They accomplish SA tasks by *Run Simulation*, i.e., executing (simulating) their current model in the center-right interface of Fig. 1, and *Make Comparison* actions, which correspond to analyzing the differences in behaviors generated by their model versus a correct expert model (as seen in the bottom row of Fig. 1). We will discuss differences in how students' use of

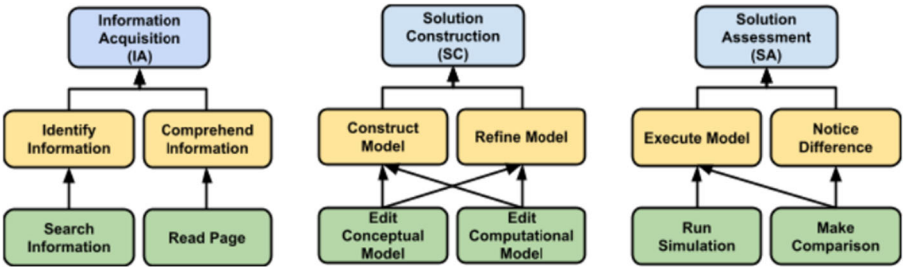


Fig. 2 The Task model for our learning-by-modeling environment

these observable actions of the task model when we compare strategies of high and low-performing groups in "Results".

The hierarchical task model helps us to map students' actions into higher-level constructs (e.g., reading a set of resource pages can be interpreted as an IA task to gather information on a specific scientific process). Frequently occurring action sequences imply that the corresponding patterns represent strategies students use in their model-building tasks (e.g., reading a set of pages (IA) to find information and using it to make changes in the model code (SC) to correct system behaviors).

Assessments of Learning Performance

The *summative* assessments include domain and CT pre- and post-tests. More specifically, we designed the acceleration and diffusion unit domain assessments to target common student misunderstandings that have been reported in the literature (Hestenes et al., 1992; Chi, 2005; Chi et al., 2012). An example of a student misunderstanding when studying diffusion processes is *particles in a liquid stop moving when the concentration levels achieve equilibrium* (Chi et al., 2012). Some of the test items in the acceleration unit were adapted from the Force Concept Inventory (Hestenes et al., 1992). We also included CT questions that focused on the mastery of fundamental programming constructs, such as loop structures and conditionals. The questions in the pre-post tests were formulated at the level appropriate for middle school students. We provide examples of the test items in the Appendix.

The *formative* evaluation of student learning consisted of the learning behaviors and performance when building the conceptual and computational science models, measured using the CA framework. These measurements helped us understand how students progressed in their learning tasks and the effectiveness of students' strategy use. Students constructed their conceptual models by selecting relevant concepts, their properties, and relevant processes to define system behaviors from a pre-populated drop-down list. We adopted a bag-of-words (BOW) representation and a simple matching similarity measure to score the students' models in relation to an expert (correct) conceptual model (Zhang et al., 2010). A larger intersection between a student's conceptual model and the expert conceptual model indicated that the model was more accurate (correct). Analogously, a student's conceptual model edit was considered to be *effective* if the similarity value between the students' model and the expert model increased.

To analyze students' computational model-building performance, we first encoded their models as abstract syntax trees (ASTs) before comparing them against the expert model using the tree edit distance (TED) (Bille, 2005; Basu et al., 2014). ASTs are tree-structured representations used in compilers to capture the syntactic structure of computer programs (Grosch & Emmelmann, 1990). They have also been used for code evaluation (Baxter et al., 1998) and studying the construction of a computer program by a student (Neamtiu et al., 2005). ASTs provide a flexible, compact, and extendable representation, where semantic information can be embedded into the tree structure to enable more in-depth analysis of a program (Rabinovich et al., 2017).

In our work, we have adapted the AST structure to capture a generalized and compact hierarchical structure of the computational models the students construct in CTSiM. Each node of the tree represents a computational block with the fields associated with the block being represented as subtrees under the primary block (Bille, 2005). We further standardize the AST representation to create canonical structures by applying semantics-preserving transformations to some of the constructs (e.g., all conditionals are expressed in the $\leq$ form) (Xu & Chee, 2003). As a result, similar programs (models) produce similar AST structures, and, therefore, similar TED scores. We have used the computed TED from the students' final models as the students' model performance score. The smaller the distance, the better the match between a pre-defined expert model and the model generated by the student.

Assessing Students Learning Behaviors

Sequential pattern mining (SPM) approaches find relevant patterns across a large number of objects, where each object is represented by an ordered sequence of items (Agrawal & Srikant, 1995). In our case, the objects represent students, and each student is represented by the ordered sequence of actions that they perform in the system. The mined patterns represent frequently occurring sub-sequences of actions across all of the students.

Two parameters characterize an SPM algorithm: (1) a support threshold, *s-support* —this specifies the number of action sequences for which a chosen pattern appears at least once in one group. In this work we set the s-support threshold to be 0.7, i.e., for a pattern to be considered at least 70% of the students perform this pattern at least once; and (2) a gap —the maximum interval of time allowed between two consecutive items of the sequence (Zaki, 2001). Since we consider strategies to be manifested from deliberately thought-out action sequences that a learner execute, we consider a long gap in time between two actions to imply that the actions are independent and not likely to be part of the same sub plan. In this work, we computed the average gap time between actions, and used this to set the gap threshold at 6 minutes. In this work, we used a second parameter when defining action sequences. These are *noise* actions, i.e., one or more irrelevant actions that a student may perform between two actions in a sequence that are relevant to the strategy (Kinnebrew et al., 2013). We use a gap value of 1, allowing for at most one noise action between two consecutive actions of a strategy.

The Differential Sequence Mining (DSM) algorithm (Kinnebrew et al., 2013) extends SPM by comparing the instance support, i.e., *i-support* value between two groups of students (e.g., high- vs. low-performers). *i-support* is defined as the average number of occurrences of a pattern across all students in a group. We hypothesize that frequent action patterns extracted from students' action sequences correspond to strategies they apply to advance their model-building and problem-solving tasks. In previous work, we have observed that students' behaviors can vary substantially, and these differences are also indicative of differences in their model-building and learning performance (Kinnebrew et al., 2014; 2017).

Study and Data

We use the results from a classroom study with CTSiM to demonstrate the students' use of learning strategies and our methods for detecting and evaluating these strategies. The participants were 52 sixth-grade students from a middle school in the United States. Among the students enrolled in the school, 9% were Asian, 26% were African American, 7% were Hispanic/Latino, 56% were White, and 2% did not provide a response. Ten percent of the students received free or reduced-price lunch.

Student work was divided into five units over 14 days (45 minutes daily) that included training and the building of three computational models. On day 1, the students took the pre-tests. On day 2, the students were introduced to agent-based modeling concepts and the CTSiM system. On days 3 and 4 (training units), students programmed software agents to draw geometric shapes. On days 5-7, the students learned to apply Newton's laws of motion (involving force, mass, acceleration, velocity, and position) to build computational models of racing cars (the acceleration unit). On days 8-10, the students learned about the conservation of momentum by modeling collisions of inelastic objects with different weights (the collision unit). On days 11-13, students learned about diffusion by constructing models of the motion of water and dye particles in a closed container (the diffusion unit). The post-tests were administered on day 14.

The data collected from the study included the summative domain and CT assessments, the conceptual and computational models that the students built, and students' logged actions in the CTSiM system. Action information, i.e., the list of logged actions was represented using a pre-defined $<action,view>$ scheme. This helped us capture the context in which students performed specific actions (Segedy et al., 2015a). The context helped us determine how proximal actions performed by the students related to one another, and this provided the basis for inferring the strategies they employed to accomplish their learning and problem-solving goals.

Specifically, our focus in this work is on detecting and evaluating the learning strategies that students develop and apply in the *acceleration* and the *diffusion* units. Assessment and log data from the collision unit were omitted from our analyses because the collision unit served as a preparation for the diffusion unit, and the major concepts associated with collisions were also covered in the diffusion unit.

Results

"Learning Performance: Domain and CT Learning Gains" presents the students' overall learning gains derived from their pre- and post-test scores. "Learning Performance: Conceptual and Computational Modeling-Building" reports students' conceptual and computational modeling performance in relation to a correct 'expert' model. These results demonstrate the overall effectiveness of the CTSiM intervention in helping students learn their science and CT, and serve as the basis for investigating **RQ3**, where we hypothesize that the use of good strategies when building science models is linked to better learning outcomes on the science and CT concepts.

"Students' Strategy Use in the Acceleration Unit" illustrates the results of applying our sequence mining algorithm to detect and interpret patterns of activity that correspond to the strategies students use in their learning and model-building tasks. These results answer **RQ 1**. The analyses conducted in "Students' Strategy Use in the Diffusion Unit" provide evidence for **RQ 2** by showing the changes in students' frequently-used strategies from the acceleration to the Diffusion unit in CTSiM. Finally, "Evaluating Strategy Use with Respect to the Learning Performance" divides the students into high (HG) and low (LG) performers based on the pre-post test learning gains and then uses the DSM method discussed in "Assessing Students Learning Behaviors" to investigate **RQ 3**, i.e., are there differences in strategies used by high and low performers? If there are differences, what do these differences represent?

Learning Performance: Domain and CT Learning Gains

The acceleration pre-post tests included nine multiple-choice questions and two short-answer questions; the diffusion test had 11 multiple-choice questions, and the CT test had five multiple-choice questions and four short-answer questions. We computed the Cronbach's α of the internal consistency for the CT test, where we believe there is an internal consistency among the test items, i.e., they test similar concepts. The two domain tests cover multiple concepts and relations from kinematics and mechanics in the acceleration test, and concepts and relations among temperature, energy, velocity, movement of particles, and collisions in the diffusion unit (see "Assessments of Learning Performance"). Therefore, we did not run Cronbach's α on the two domain tests.

The moderately high value of Cronbach's α for the CT test, 0.77, indicates that the test items have high internal consistency with each other. This is because the test covers questions related to basic computing constructs, such as loops and conditionals, as well as CT practices like decomposition and debugging, which require interpretation and application of the basic constructs (Weintrop et al., 2016). However, like the domain constructs, we designed the CT questions to achieve coverage instead of internal validity.

Kolmogorov-Smirnov tests run on the students' pre-post test scores showed that the scores were not normally distributed. Therefore, we used the non-parametric Mann-Whitney U-Test to determine the significance of the students' *pre-to-post* learning gains. In addition, taking into account the non-normality of the data and following the suggestion from Rosenthal et al. (1994), we calculated the effect size of the learning gains as $\frac{Z}{\sqrt{N}}$) instead of using Cohen's d.

Table 1 lists the median, the 25$^{\text{th}}$ percentile, and the 75$^{\text{th}}$ percentile of the students' pre-test and post-test scores. It also reports the $p-$values for the test of significance and the effect sizes. The max scores achievable for the three tests were 24, 12, and 34 points, respectively. In all three tests, students showed significant improvements in their pre- to post-test scores, with moderate effect sizes for the acceleration and CT test gains (0.43 and 0.57, respectively), and a large effect size for the diffusion test (0.9). These results indicate that the intervention helped students improve their understanding of both the domain and CT skills and practices.

Table 1 Pre-post learning gains

Measure	Pre-test Scores (Median, [25%, 75%])	Post-test Scores (Median, [25%, 75%])	p-value	Effect Size
Acceleration	13.5 [11, 15.5]	16 [13, 18]	0.01	0.43
Diffusion	4 [2, 5]	5 [4, 7]	< 0.0001	0.90
CT	13.5 [10.5, 13.5]	19.5 [18, 22.5]	0.002	0.57

Learning Performance: Conceptual and Computational Modeling-Building

In CTSiM, the **conceptual** modeling activity (corresponding to the top-right screenshot in Fig. 1) requires that the learner identify the important components in the system as agents. Along with this, the learner needs to identify the relevant properties and behaviors of the agents in the system. Students are provided with a set of constructs that include the relevant ones plus additional extraneous constructs, and this allows us to evaluate the students' abilities to pick the accurate (i.e., they include all relevant constructs) and precise models for the needed agents (i.e., they do not include irrelevant constructs). The constructs include agent properties, additional variables, and behaviors for each agent. The students also choose the agent behaviors with their corresponding input-output parameters. In addition, they choose the environment variables that may influence the model behaviors.

By comparing students' conceptual models to an expert-defined set of needed agents and their associated constructs, we can identify situations where students miss variables or behaviors, or if the student adds unnecessary constructs to their model. For example, in the task for modeling accelerating cars, each car is modeled as an agent with relevant properties, such as the traction force generated by the car's engine and its mass, that are required to define the car's motion. On the other hand, properties, such as the car's cost and color are not directly relevant to the equations that describe the car's motion.

For this paper, we evaluated learners' **conceptual** models by comparing the student-generated models with an expert model using the BOW representation along with the similarity measure (note: dissimilarity/distance $= 1-$ similarity). Figure 3

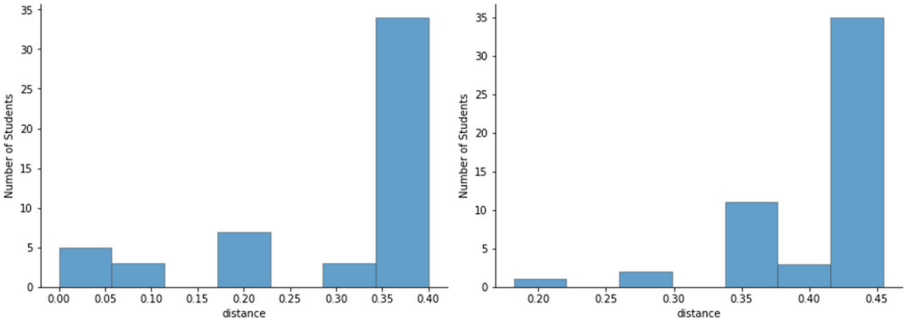


Fig. 3 The final extraneous conceptual model distances (normalized) in the acceleration unit (left) and the diffusion unit (right)

shows the number of students that had extraneous components in their models when compared against the expert (correct) conceptual model. Students who had more correct conceptual components in their models proportionally added less irrelevant and unnecessary components to their models. We use the addition of irrelevant components as a proxy for a lesser understanding of the domain concepts, and the conceptual model score reflects this fact. The average final conceptual model distance across all students was 3.11 ($SD = 1.35$) for the acceleration unit and 9.60 ($SD = 1.32$) for the diffusion unit.

We also computed the students' final **computational** model distance to the structure of the expert model using the tree-edit distance (TED) metric.The histogram in Fig. 4 shows the students' computational model score on the x-axis, and the number of students who had extraneous model constructs for the two units. The plot on the left represents the tree edit distance (TED) for the acceleration unit and the plot on the right is the edit distance for the diffusion unit. Students whose model score distance $= 0$ generated models that matched the expert model. The greater the edit distance, the more extraneous or incorrect constructs were included in the model. Some student models contained unnecessary procedural components in their agent behaviors. In most cases, these additional elements did not affect the agent's behaviors (for example, having a superfluous `setColorOfCar` procedure in the acceleration unit did not influence the motion of the car). In these cases, the student model often generated the same behaviors as the expert model, even though the TED between the two models was not 0.

Unlike the conceptual modeling task, students with more complete computational models also tended to have extraneous components in their models. A possible explanation for this could be that students initially had difficulties with their models, and, therefore, started by adding a number of extraneous components to their models. But, they did not remove them at the end. In other words, their focus seemed to be on getting a model that generated the correct behaviors, and not on building the most compact or precise model to generate those behaviors. In the acceleration unit, the average final distance between the student models and the expert model was 7.58 ($SD = 6.34$), and the average final distance between the student models and the expert model in the diffusion unit was 22.8 ($SD = 8.99$). The diffusion models were

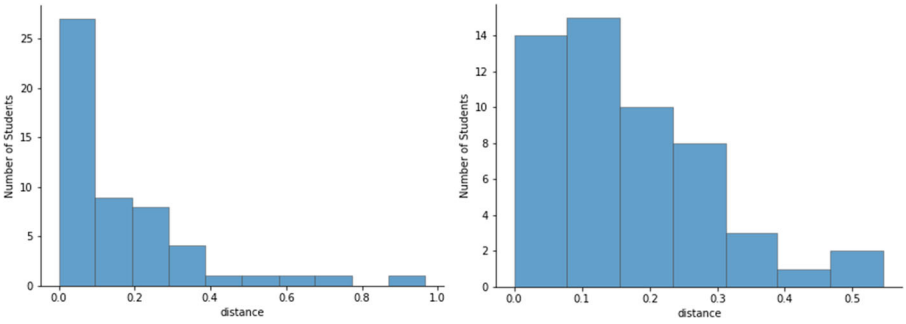


Fig. 4 The final extraneous computational model distances (normalized) in the acceleration unit (left plot) and the diffusion unit (right plot)

less accurate than the acceleration models, and this may be attributable to the greater complexity of the model.

Student's Learning Strategies

To investigate students' strategy use in CTSiM and how strategy use evolved across the two modules, we mined students' logged actions in the acceleration and the diffusion units using an SPM algorithm (Zaki, 2001). As discussed earlier, this analysis helped us answer **RQ 1**. Student actions were characterized by five specific actions that we explained in greater detail in "The CTSiM OELE": (1) IA_read, (2) SC_conceptual, (3) SC_computational, (4) SA_run, and (5) SA_compare. The links from these five actions to their higher-level categories appear in the CTSiM task model (see "The Task and Strategy Models of CTSiM")[2].

To simplify the derived pattern representations, we adopted notation from *automata theory* (Hopcroft et al., 2001), and replaced multiple consecutive occurrences of the same action with a $^+$ superscript symbol; e.g., IA→IA→IA may be represented as IA$^+$. We also annotated SC actions with additional information by adding a suffix **-EFF** to indicate that the action was effective (i.e., it reduced the student's model BOW distance for the conceptual, and the TED distance for the computational models with respect to corresponding expert models). The suffix **-INEFF** was used when a model editing action resulted in a decrease in a model score.

For example, a pattern **SC_computational-EFF**$^+$ → **SA_compare** indicates that the student made a series of consecutive effective SC edits on their computational model, and then ran a simulation to compare the behaviors of their model against the expert model.

The following parameter settings were used for the sequence mining algorithm: (1) the threshold value of support for considering a pattern was set at 0.7, implying that a pattern needed to occur in at least 70% of the students' action sequences; and (2) the maximum noise gap between two actions was set at 1.

Students' Strategy Use in the Acceleration Unit

Table 2 lists the frequent patterns mined with the action log data in the acceleration unit by our SPM algorithm. The transitions between actions in a pattern are linked by →, and the patterns are ordered in descending order of their *s-support* values. We also report the *i-support* values for all of the sequences that met the *s-support* threshold.

An immediate observation from the derived action patterns shows that the students tended to read several resource pages (IA) in succession, and then switched to solution construction (SC) actions, performing a number of model-building actions. This approach is indicative of a "depth-first" approach to acquiring information (IA) and then building model constructs (SC). In contrast, one might have expected other students to adopt a more targeted back-and-forth IA → SC strategy for model-building,

[2]These actions are labeled by combining the generic action types (IA, SC, and SA) and the observable data logged in the OELE (see also Fig. 2)

Table 2 Action patterns mined from log data in the acceleration unit

	Pattern	S-Support	I-Support	Strategy Type
1	SA_run→SC_computational-EFF^{+}	0.94	5.13	**SASC**
2	SC_computational^{+}→SA_run^{+}	0.90	4.48	**TK**
3	SA_run^{+}→SC_computational-EFF^{+}	0.90	3.48	**SASC**
4	IA_read^{+}	0.85	6.73	**MIA**
5	SC_computational-EFF→SC_conceptual^{+}→SC_computational	0.83	2.12	**MTPR**
6	SA_run→SC_computational-EFF^{+}→SA_run	0.83	3.21	**SASC**
7	SC_computational→SC_conceptual^{+}→SC_computational-EFF	0.81	2.10	**MTRP**
8	SC_computational→SA_run→SC_computational-EFF^{+}	0.81	2.63	**SASC**
9	SC_computational-EFF→SA_run→SC_computational-EFF^{+}	0.79	2.13	**SASC**
10	IA_read→SC_computational-EFF^{+}	0.77	1.48	**IASC**
11	SC_computational-EFF→IA_read	0.71	1.60	**SCIA**

which may correspond to a *building in parts* approach that students may apply when building more complex models.

Falling back on our definition of strategies as students' *conscious and controllable action sequences to facilitate and enhance task performance*, we provide an interpretation for the discovered patterns as learning strategies in the context of the CTSiM environment.

1. **Multiple information acquisition (MIA)**. Pattern #4. Students used this as a *depth-first* strategy to acquire relevant knowledge from the domain and CT resources to support their model-building. We assume that students used this strategy early on to gain the information they needed to support their model-building tasks. Not surprisingly, this pattern (#4) was performed by 85% of the students, and it was also the most frequent pattern (average of 6.73 occurrences per student in their activity sequences), supporting the conjecture that the students started as naive learners and depended on the resources a lot to gain relevant domain and CT knowledge.
2. **Support solution construction with information acquisition (IASC)**. Pattern #10. This pattern contrasts with the depth-first MIA strategy. Presumably, students found specific knowledge and information that they applied to their model-building tasks. In pattern #10, the IA action was followed by one or more effective computational model edits. The average use of this pattern per student was low ($\approx$ 1.5 per student).
3. **Check the constructed model by reading resources (SCIA)**. Pattern #11. This is the flipped version of the IASC strategy. In this case, students strategically re-read the resource page to check that the SC actions they generated were correct as they understood it. The number of occurrences of this pattern per student was low ($\approx$ 1.6).
4. **Tinkering (TK)**. Pattern #2. When *tinkering*, students adopted a trial-and-error strategy to complete a task. The trial-and-error search was not completely

random and can imply some just-in-time planning or narrowing down by elimination strategies (Berland et al., 2013). In CTSiM, tinkering activity can be detected as interweaved and repeated solution construction (SC) and solution assessment (SA) actions. Most students ($\approx$ 90%) used this pattern, on an average of $\approx$ 4.5 times per student. It also shows that novice learners adopt incremental methods to support their model-building.

5. **Gain understanding with multiple representations (MTRP)**. Patterns #5 and #7. In this case, students switched between the two modeling representations, implying that they were applying a form of the *model-building in parts* strategy. In pattern #5, students started with effective computational edits but made errors in subsequent conceptual and computational edits when extending their model. For pattern #7, this was reversed. The students initially made incorrect edits, but then made a correct computational edit. On average, this strategy was executed approximately twice by each student. Our analysis indicates two possible interpretations: (1) students may be leveraging a *model-building in parts* strategy effectively to work on a complex problem (Basu et al., 2017) and (2) students were ineffective in applying *model-building in parts*, and went back and forth between their conceptual and computational models trying to determine what agent properties and behaviors they may need to add to complete their model. The latter interpretation is better explained in our differential sequence mining results, where we differentiate between the HG and LG student behaviors.
6. **Revising Model (SASC)**. Patterns #1, #3, #6, #8, and #9. When students evaluated their solutions (computational models) by running a simulation, analyzing the results generated may have provided information that helped them improve their current model. The SA action(s) constitute checking and assessing solutions, and students may have used the information generated from the analyses to correct errors in their models. Patterns #1 and #3 represent the use of this strategy with the students sometimes making multiple comparisons to check their model behaviors, and sometimes making multiple edits to improve their model. A more advanced variation of this strategy is represented by pattern #6. Here students followed up on the SC steps by re-running, and, therefore, re-checking their model for additional errors. Patterns #8, and #9 represent a quick check after a model-building action, and then a sequence of model-building actions.

These results demonstrate that students developed and applied different strategies that combined their reading, model-building, and model-checking activities to support their model-building tasks. The build and assess, and assess and rebuild strategies were quite frequent, along with tinkering, which is a widely used mechanism that novice programmers use for learning and understanding programming constructs (Berland et al., 2013; Bers et al., 2014).

Students' Strategy Use in the Diffusion Unit

To study how students' use of strategies evolved over time, we applied our sequence mining methods to the logged CTSiM action sequences in the diffusion unit, which

Table 3 Differences in action pattern use between the diffusion and acceleration units

	Pattern	Strategy Type	I-Support (Acceleration)	I-Support (Diffusion)	Effect Size
1	IA_read→SC_computational-EFF$^+$	IASC	1.48	0.17	1.17
2	SC_computational-EFF→IA_read	SCIA	1.60	0.08	1.03
3	IA_read$^+$	MIA	6.73	1.81	0.97
4	SC_computational→SC_computational-EFF$^+$→SA_run$^+$	TK	2.62	0.37	1.43
5	SC_conceptual$^+$→SC_computational-EFF$^+$	MTRP	0.76	2.03	1.34

students worked on after completing the acceleration and collision units. The analysis showed small shifts in the strategies that students used. Table 3 lists the action patterns that were different in the two units. We looked for large effect sizes in the difference between the *i-support* values for the two units. To control for type I errors, we applied a False Discovery Rate correction (Glickman et al., 2014) on the hypothesis test results of the different patterns and confirmed that these differences were statistically significant[3].

Other than the MTRP (modeling in parts with multiple representations) strategy, the use of all other strategies reduced considerably from the acceleration to the diffusion unit. For example, the use of the MIA, IASC, and SCIA strategies reduced considerably (effect sizes of 0.97, 1.03, and 1.17, respectively). This change may be attributed to the students' increased familiarity with the learning environment and the resources, e.g., the CT resources were the same for all units. Students were less likely to access resources on topics they had become familiar with, which may explain the drop in the *i-support* values for the MIA, IASC, and SCIA strategies. However, the fact that this led to larger conceptual and computational model distances for the diffusion model (see "Learning Performance: Conceptual and Computational Modeling-Building"), implies that the students would have benefited from continuing to apply these strategies, especially by reading the resources to better understand the diffusion process.

The use of the model-building in parts (MTRP) strategy increased considerably (effect size $=$ 1.34). Students seemed to go back and forth between their conceptual model (to get the right framing) and the computational model (to implement behaviors correctly) for the more complex diffusion model. In previous work, we have shown that greater use of the MTRP strategy, interpreted as building the model in parts, can be linked to more successful model-building and higher pre-post learning gains (Basu et al., 2017). Interestingly, students' use of the tinkering strategy became more refined and more effective, i.e., they were followed by more correct computational model edits. However, the TK strategy was used less than in the acceleration unit (effect-size $\approx$ 1.4). The complexity of the diffusion model may

[3]Most of the computed p-values were < 0.0001. Therefore, the correction did not change the results

Table 4 Confusion matrix of the breakdown of HG and LG students vs. their post-test performance

n=52	post-test low	post-test high
gain low	18	6
gain high	7	21

have affected students' performance (model scores) and their ability to apply more advanced strategies.

A possible takeaway from these results is that novice learners need more support to become effective strategy users. When faced with difficulties that impede their progress in model building, students tend to use more trial-and-error and tinkering strategies. However, a better approach is to decompose the complex tasks into smaller sub-tasks to manage this complexity. Therefore, timely feedback in these situations (as we demonstrated in Basu et al. (2017) may help students overcome their difficulties and make more progress in their learning and model-building tasks. Overall, this helps us answer **RQ 2**, i.e., students' use of learning strategies evolved as they worked on multiple modeling tasks.

Evaluating Strategy Use with Respect to the Learning Performance

Results presented in "Learning Performance: Domain and CT Learning Gains" showed that the overall learning gains in the acceleration and diffusion unit were significant (effect sizes of 0.43 and 0.9, respectively). Leveraging these results as well as the identification and interpretation of common strategies through SPM (see "Student's Learning Strategies"), and to answer **RQ 3**, we study the relationship between the use of strategies and students' performance measured by (1) their overall learning gains, and (2) their ability to build correct models.

We divided the 52 participants in the classroom study into two groups by their performance: (1) a high learning-gain group (HG, $n = 28$) and (2) a low learning-gain group (LG, $n = 24$). One of the primary results we show in this paper is the link between the use of strategies and learning. To demonstrate this, we chose the pre-post learning gains for the HG-LG division. We used a median split on the learning gains derived from the pre-post test scores.[4] We also show in Table 4 that if we performed a median split on the post-test scores, 21 students with high post-test scores also had high learning gains. Similarly, 18 students with low post-test scores had low learning gains. A much smaller number, 6, students had high post-test test scores but low learning gains. Correspondingly, 7 students were below the median scores on the post-test, but their learning gains were high.

[4]The number of HG students is slightly larger because four students had the same learning gain as the median and were assigned to HG.

Table 5 Comparison of HG and LG student's performance and behavior measures

Measure	HG (Median [25%, 75%])	LG (Median [25%, 75%])	*p*-value	Effect Size
Diffusion-gain	4 [3, 5]	0.5 [-1.25, 1]	<0.0001	0.80
CT-gain	4.75 [-0.125, 8.125]	2.5 [-1, 6.25]	0.02	0.29
conceptual distance (Acc.)	4 [3.5, 4]	4 [2, 4]	0.08	0.20
computational distance (Acc.)	5.5 [2, 10]	5 [3, 11]	0.24	0.10
conceptual distance (Diff.)	10 [9.5, 10]	10 [10, 10]	0.45	0.02
computational distance (Diff.)	19 [16, 26]	26 [17.5, 31]	0.15	0.15
Measure	HG time distribution (*M, SD*)	LG time distribution (*M, SD*)	*p*-value	Effect Size
Read	4% (7%)	3% (9%)	0.28	0.08
Conceptual Edit	24% (11%)	22% (9%)	0.56	0.03
Computational Edit	32% (11%)	33% (11%)	0.59	0.03
Run (test)	10% (9%)	8% (7%)	0.38	0.04
Comparison	14% (9%)	13% (7%)	0.87	0.06
dle	16% (7%)	20% (9%)	0.15	0.14

The upper half of Table 5 summarizes the differences in the performance measures (learning gains in diffusion and CT, and the conceptual and computational modeling scores for the acceleration and diffusion units) for the HG and LG students. Mann-Whitney U-tests were performed to check if the differences were statistically significant. Table 5 shows that HG students had significantly higher learning gains in the diffusion and CT pre-post tests. However, although the computational modeling scores for the diffusion unit showed a meaningful imbalance in the baseline covariance with effect sizes > 0.1 (Austin, 2009), the modeling score differences between the HG and LG groups were not statistically significant.

To understand the performance gap in the test scores, we also computed the time spent on the different CTSiM activities for the HG and LG students. The bottom section of Table 5 shows the percentage of time that HG and LG students spent in all five learning activities and their estimated idle time percentage. Overall, the large p-values and small effect sizes indicate that the HG and LG students did not differ by much in the time spent on the different learning and model-building activities. The total time logged in the system by students in each group was also similar (HG $mean = 2746$ seconds, LG $mean = 3124$ seconds, $p = 0.14$).

As a next step, we compared the strategies used by the two groups to see if they may explain the difference in the students' pre-post learning gains. To examine the differences in strategy use by the HG and LG students, we ran the differential sequential mining algorithm (Kinnebrew et al., 2013) to extract the patterns that showed sufficient differences between the two groups. Table 6 presents the mined patterns for which the HG and the LG groups had statistically significant differences in the

Table 6 Patterns with different support among HG and LG students

	Pattern	Strategy Type	I-Support (HG)	I-Support (LG)	Effect Size
1	SA_run→SC_computational-EFF$^+$→SC_conceptual-EFF	SASC	1.14	0.58	0.55
2	SA_run→SC_computational-EFF$^+$	SASC	0.71	0.38	0.50
3	SC_conceptual-EFF$^+$→SC_conceptual-EFF$^+$	MSC	1.32	0.71	0.63
4	SC_computational$^+$→SA_compare→SC_computational$^+$	TK	0.14	0.88	1.06
5	SC_computational-EFF$^+$→SC_computational→SC_conceptual$^+$→SC_computational	MTPR	0.43	1.04	0.85
6	SC_computational$^+$→SC_computational-EFF$^+$→SA_compare→SC_computational	TK	0.18	0.83	0.88
7	SC_computational-EFF$^+$→SC_computational$^+$	MSC	0.39	1.17	0.97

i-support values for the diffusion unit. Similar to the results presented in Table 3, we applied the False Discovery Rate adjustment to the *p*-values and verified that these patterns were still statistically significant.

The patterns reported in Table 3 are divided into two groups: (1) patterns that HG students performed more frequently than LG students, (2) patterns that LG students performed more frequently.

The effect sizes for the two SASC patterns (#1 and #2) indicate that the HG students were more successful in making corrections to their models after comparing their model behavior to the expert model behavior. The SASC patterns capture students debugging behaviors. Whereas the use of the SASC strategy did not seem to result in a clear difference in the model scores between the two groups, the better debugging behaviors may have led to the better domain and CT learning gains. On the other hand, patterns #4−#6, which were performed more by the LG students indicate continued unproductive tinkering (TK) behavior and the use of multiple representations during model-building (MTPR). These students SC actions were mostly not EFF (effective)) in improving their models. Therefore, the LG students did not seem to learn the relevant domain and CT knowledge while working on their model-building tasks.

Finally, both the HG and the LG students performed multiple SC behaviors (patterns #3 and #7) indicating a depth-first strategy to model-building. Pattern #3, which was performed more by the HG, shows that they were better at conceptualizing their models than the LG students. Pattern #7 shows that the LG students were more likely to make incorrect computational model edits even after they had made a few correct ones. Deficiencies in their domain and CT knowledge may have affected their

model-building performance, and their inability to correct their models after running SA actions (Patterns #4−#6 as opposed to Patterns #1 and #2) made it harder for them to build correct models and learn from their model-building tasks. Similar issues of depth-first computational modeling behaviors have been reported in Grover et al. (2016).

Discussion and Conclusions

This work extends the existing research on strategy analysis in open-ended learning environments (OELEs), and in particular, environments that are geared to the *learning-by-modeling* paradigm. We have provided a systematic approach for the modeling, detection, and interpretation of students' learning strategies as they work on their knowledge acquisition, model building, and problem-solving tasks. It adds value to recent research that applies data-driven approaches to study strategy use from traces of students' 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 most of the reported 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).

In contrast, the focus in this work has been on studying middle school students' use of strategies in learning-by-modeling (that is how we use it elsewhere in the paper) and problem-solving environments. Unlike undergraduate students, middle school students (10-12-year-olds) are novices, just beginning to learn science and computational concepts in a systematic way. Building and analyzing conceptual and computational models in science domains by combining knowledge acquisition, computational model building, and model checking is a complex task for these students. As our study has demonstrated ("Students' Strategy Use in the Acceleration Unit"), not all students learn effective strategies as they are working in our environments, and the ineffective strategy use affects students' learning of domain and computational constructs through model-building tasks (see Table 1 and Figs. 3 and 4). But students become better at using strategies as they progress through the curriculum, and it is interesting to study how they develop and apply strategies that span domain and CT learning to succeed in their tasks ("Students' Strategy Use in the Diffusion Unit").

In summary, our current work provides (1) an operational definition of learning strategies in a learning-by-modeling environment where students are involved in three primary activities: information acquisition, solution construction, and solution assessment, and (2) the design and implementation of a framework to model, measure, and evaluate students' learning strategies. A significant finding in our analysis is that high-performing students were better at applying the SASC strategy to detect and

correct errors in their models by comparing their model behaviors against expert (i.e., correct) model behavior. Furthermore, tracking students' behaviors from the acceleration to the diffusion unit showed how high-performing students' debugging and learning strategies evolved from tinkering to systematic checking. High-performers interspersed model construction (SC) and model assessment (SA) actions, making it easier for them to detect errors as they built their models in parts. On the other hand, low-performing students continued to use tinkering strategies in the diffusion unit. Evidence shows that there are correlational (not causal) links between systematic debugging ability and better understanding of the domain and CT knowledge in high performing students. We would like to study these relations in greater detail in future work.

This finding leads us to believe that students who struggle to apply effective learning strategies may benefit from adaptive scaffolding to aid them in their model-building and learning tasks. Initial studies with adaptive feedback on a few strategies in the CTSiM system helped students become better learners (Basu et al., 2017). In future work, we will develop targeted scaffolding and feedback to help students develop effective model-building and model-debugging strategies. An important component of this work will be to tease apart students' difficulties with the domain concepts and computational constructs and practices, and then guide them to using strategies that will help them combine concepts and practices across these two domains.

As discussed in the literature review, a significant amount of work has been done in defining and characterizing strategies and explaining them in the context of students' self-regulated learning behaviors. However, we also indicated limitations in the existing literature on learning strategies. In particular, we discussed three in "Problem Statement": (1) the grounding problem; (2) the generalizability problem; and (3) the measurement problem.

In this work, we addressed the *grounding problem* by developing an operational definition of strategies in the context of tasks that students perform when constructing computational models to learn about scientific processes. We also addressed the *measurement problem* by representing the student behaviors in a hierarchical representation using Coherence Analysis and then apply sequential pattern mining and differential sequential pattern mining approaches (Kinnebrew et al., 2013; Zaki, 2001) to understand students' strategic behaviors in the context of the tasks they were performing. The sequence mining approach differs from other related work that has used clustering (e.g., Segedy et al. 2015a), MANOVA (e.g., Gillies 2003), and Hidden Markov Model methods (e.g., Biswas et al., 2010; Boyer et al., 2011) to characterize learning behaviors and strategies. In the work we present, our analysis of strategic behaviors goes beyond discovering and labeling strategies as sequences of actions. Rather, we have linked students' use of strategies to their success in accomplishing their model building and learning of domain knowledge.

Regarding the *generalizability problem*, we have shown examples of generalizability of this approach by demonstrating its applicability in two learning domains, Betty's Brain (past work) and CTSiM (current work). Currently, we are extending the approaches presented in this paper to other computational modeling environments, such as C2STEM (Hutchins et al., 2020b) and SPICE (McElhaney et al., 2020;

Zhang, 2020). We have also analyzed students' strategy use when they learn cybersecurity concepts in a network programming environment (NetsBlox) (Yett et al., 2020).

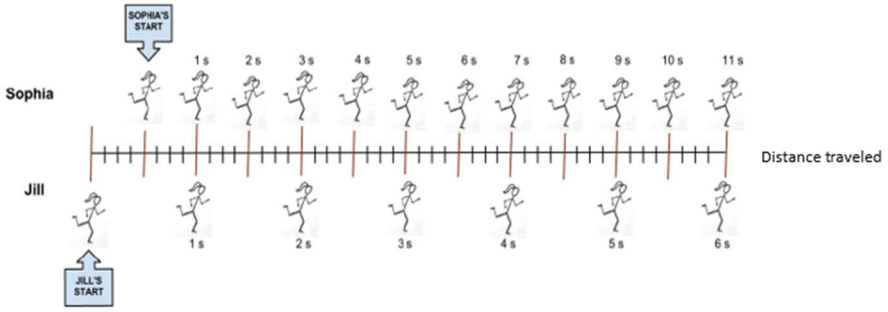
More generally, we would like to scale up our work and encourage other researchers to adopt this framework in problem-based learning environments (Hmelo-Silver, 2004). This would require applying cognitive task analysis (Schraagen et al., 2000) and evidence-centered design (Mislevy et al., 2003) approaches to create task subtask hierarchies as a framework for interpreting and analyzing students' activity sequences, much like we have done in this paper and in current work with other systems (Hutchins et al., 2020a; Kinnebrew et al., 2017; Zhang, 2020). Similarly, the learning environment has to be designed to capture fine-grained activity data and information that helps to interpret these actions in the context of relevant subtasks and subproblems (Bernacki, 2017; Hadwin et al., 2007; Segedy et al., 2015a). In addition to sequence mining (Zaki, 2001; Kinnebrew et al., 2014), a number of analytics and machine learning methods have been applied for analyzing activity trace data, e.g., Markov chain modeling (Zhang, 2020), Hidden Markov models (Biswas et al., 2010; Boyer et al., 2011; Fincham et al., 2018), process mining (Juhaňák et al., 2019; Maldonado-Mahauad et al., 2018), clustering methods (Kovanović et al., 2015; Segedy et al., 2015b; Zhang et al., 2017), knowledge tracing (Pelánek, 2017; Piech et al., 2015), and path analysis (Bernacki et al., 2012; Zhang et al., 2020). In other words, data-driven methods for analyzing students' strategic learning behaviors is a very rich and fertile field that can contribute greatly to next-generation intelligent learning environments.

Students' development of productive strategies is a key factor in succeeding in today's technologically advanced workforce needs. Computer-based learning environments, and specifically OELEs, provide the means for students to develop and practice important learning and problem-solving strategies. Leveraging this framework, future research can explore students' strategy development and its' impact on learning and problem solving across different domain topics and problem solving contexts. There are also fertile opportunities to exploit the identification and interpretation of students' strategies for the development of adaptive feedback to support their learning and problem solving, especially when they face difficulties. Developing online methods can also aid teachers in noticing and responding to students' conceptual understanding and problem-solving in context as part of their classroom instruction.

Appendix

Example Domain and CT question items used in the pre-post tests. Figure 5 shows a test item in the acceleration pre-post test. It was adapted from the Force Concept Inventory (Hestenes et al., 1992). Figure 6 is question 6 in the diffusion pre-post test adapted from Chi et al.'s work targeting the common misunderstanding of particle diffusion (Chi et al., 2012). Figures 7 and 8 target on the CT concepts of variable, sequence, Boolean logic, and conditionals. The items in the CT test come from Basu et al.'s work (Basu et al., 2017).

1. Sophia and Jill start running at the same time and run in the same direction. The diagram below shows Sophia's and Jill's starting positions. Their positions at every second after they start are also marked on the diagram. Each small hash mark represents one meter, and "s" means *seconds*.



Circle your answers

a) Who is running faster

(A) Sophia

(B) Jill

b) Do Sophia and Jill accelerate as they run?

Sophia accelerates: (A) YES (B) No

Jill accelerates: (A) YES (B) No

c) Whose acceleration is greater?

(A) Sophia

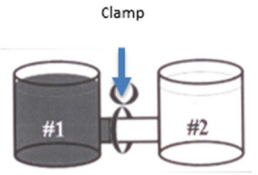
(B) Jill

(C) Both accelerate at same rate

(D) Neither accelerate

Fig. 5 Question 1 in the acceleration unit pre-post test

Suppose you have two beakers connected by a short tube with a clamp (see below). Beaker #1 contains a highly concentrated solution of darkly colored blue dye and water. Beaker #2 contains only water.



6. After the clamp is removed, and you see the flow of dye from Beaker #1 to #2, which statement best describes the behavior of the molecules?

(A) Dye molecules only move from high concentration (Beaker #1) to low concentration (Beaker #2).

(B) The molecules need to reach equilibrium, so some dye molecules move back into Beaker #1 to keep a balance.

(C) All the molecules move and collide randomly, so a dye molecule could go back into Beaker #1 by chance.

(D) The dye molecules can only move in the direction of the flow that we see from Beaker #1 to #2.

Fig. 6 Question 2 in the diffusion unit pre-post test

2. Consider the following program

If (quiz score is equal to 10)
Then: Get the "You're a pro" sticker
Else: _____
If (quiz score is greater than 7)
Then: Get the "Good job" sticker
Else: _____

Bill gets a score of 9 points on the quiz, and Janet gets a score of 10. What stickers should Bill and Janet receive based on the above program?

a. Bill: "Good job" sticker; Janet: "You're a pro" sticker
b. Bill: "Good job" sticker; Janet: "Good job" sticker
c. Bill: "Good job" and 'You're a pro" stickers; Janet: "You're a pro" and "Good job" stickers
d. Bill: "Good job" sticker; Janet: "Good job" and "You're a pro" stickers

Fig. 7 Question 2 in the CT pre-post test

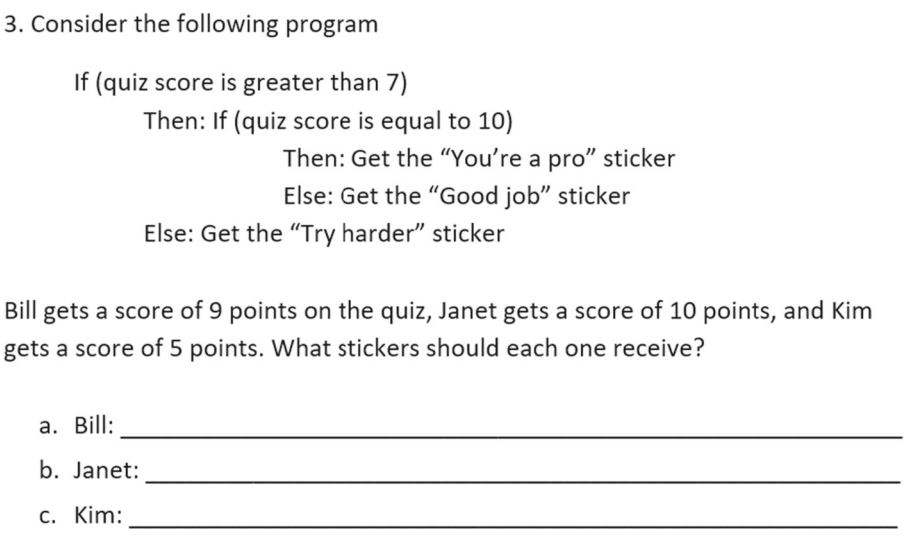


Fig. 8 Question 3 in the CT pre-post test

Acknowledgements This material is based upon work supported by the National Science Foundation under Grant 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.

The authors would like to thank Douglas Clark, Satabdi Basu, Ashlyn Pierson, Jenna Peet, and Naveeduddin Mohammed for their contribution in conceptualizing the research, adapting the pre-post assessment questions, providing educative materials, and developing the CTSiM learning environment. A different and abbreviated version of the present work using the same classroom study data has been published as a conference poster (Zhang & Biswas, 2018). We extended both the theoretical framework and analytical methods described in the poster paper, and have included a significant amount of new results in the present work.

Author Contributions Conceptualization: Gautam Biswas; Methodology: Gautam Biswas; Software: Ningyu Zhang; Investigation: Ningyu Zhang, Nicole Hutchins; Formal analysis: Ningyu Zhang, Gautam Biswas; Writing - Original Draft: Ningyu Zhang; Writing - Review & Editing: Gautam Biswas, Nicole Hutchins; Supervision: Gautam Biswas.

Funding This research was supported by a National Science Foundation Cyberlearning Grant #1441542

Data Availability Data and materials created for this research are available upon request. Please direct all inquiries to the corresponding author. Access to the CTSiM software can be required at https://wp0.vanderbilt.edu/oele/software/.

Code Availability The code created for this research is available upon request. Please direct all inquiries to the corresponding author.

Declarations

Conflict of Interests No potential conflicts of interest.

References

Agrawal, R., & Srikant, R. (1995). Mining sequential patterns. In *Proceedings of the eleventh international conference on data engineering* (pp. 3–14). https://doi.org/10.1109/ICDE.1995.380415.

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

Alesandrini, K. L. (1981). Pictorial–verbal and analytic–holistic learning strategies in science learning. *Journal of Educational Psychology*, *73*(3), 358.

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

Austin, P. C. (2009). Balance diagnostics for comparing the distribution of baseline covariates between treatment groups in propensity-score matched samples. *Statistics in Medicine*, *28*(25), 3083–3107.

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

Azevedo, R., Harley, J., Trevors, G., Duffy, M., Feyzi-Behnagh, R., Bouchet, F., & 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* (pp. 427–449). Springer.

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

Basu, S., Dukeman, A., Kinnebrew, J. S., Biswas, G., & Sengupta, P. (2014). Investigating student generated computational models of science. In *International Conference on Learning Sciences Boulder, CO: International Society of the Learning Sciences*.

Basu, S., Sengupta, P., & 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.

Basu, S., Biswas, G., Sengupta, P., Dickes, A., Kinnebrew, J. S., & Clark, D. (2016). Identifying middle school students' challenges in computational thinking-based science learning. *Research and Practice in Technology Enhanced Learning*, *11*(1), 13.

Basu, S., Biswas, G., & 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.

Baxter, I. D., Yahin, A., Moura, L., Sant'Anna, M., & Bier, L. (1998). Clone detection using abstract syntax trees. In *Proceedings. International Conference on Software Maintenance (Cat. No. 98CB36272)* (pp. 368–377). IEEE.

Beaudoin, L., & Winne, P. (2009). Nstudy: An internet tool to support learning, collaboration and researching learning strategies. In *Canadian e-Learning conference*.

Berland, M., Martin, T., Benton, T., Petrick Smith, C., & Davis, D. (2013). Using learning analytics to understand the learning pathways of novice programmers. *Journal of the Learning Sciences*, *22*(4), 564–599.

Bernacki, M. L. (2017). Examining the cyclical, loosely sequenced, and contingent features of self-regulated learning: Trace data and their analysis. In *Handbook of self-regulation of learning and performance* (pp. 370–387). Routledge.

Bernacki, M. L., Byrnes, J. P., & Cromley, J.G. (2012). The effects of achievement goals and self-regulated learning behaviors on reading comprehension in technology-enhanced learning environments. *Contemporary Educational Psychology*, *37*(2), 148–161.

Bers, M. U., Flannery, L., Kazakoff, E. R., & Sullivan, A. (2014). Computational thinking and tinkering: Exploration of an early childhood robotics curriculum. *Computers & Education*, *72*, 145–157.

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

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

Biswas, G., Jeong, H., Kinnebrew, J. S., Sulcer, B., & ROSCOE, R. (2010). Measuring self-regulated learning skills through social interactions in a teachable agent environment. *Research and Practice in Technology Enhanced Learning*, *5*(02), 123–152.

Biswas, G., Segedy, J. R., & 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.

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

Boyer, K. E., Phillips, R., Ingram, A., Ha, E. Y., Wallis, M., Vouk, M., & Lester, J. (2011). Investigating the relationship between dialogue structure and tutoring effectiveness: a hidden markov modeling approach. *International Journal of Artificial Intelligence in Education*, *21*(1-2), 65–81.

Chao, P. Y. (2016). Exploring students' computational practice, design and performance of problem-solving through a visual programming environment. *Computers & Education*, *95*, 202–215.

Cheng, B., Ructtinger, L., Fujii, R., & 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., & Chase, C.C. (2012). Misconceived causal explanations for emergent processes. *Cognitive Science*, *36*(1), 1–61.

Cloude, E. B., Taub, M., & Azevedo, R. (2018). Investigating the role of goal orientation: metacognitive and cognitive strategy use and learning with intelligent tutoring systems. In *International conference on intelligent tutoring systems* (pp. 44–53). Springer.

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

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. pp. 347–379.

Donker, A., de Boer, H., Kostons, D., van Ewijk, C. D., & van der Werf, M. (2014). Effectiveness of learning strategy instruction on academic performance: A meta-analysis, (Vol. 11 pp. 1–26). https://doi.org/10.1016/j.edurev.2013.11.002. http://www.sciencedirect.com/science/article/pii/S1747938X13000420.

Fincham, E., Gašević, D., Jovanović, J., & 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.

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

Garner, R. (1988). 5 - verbal-report data on cognitive and metacognitive strategies. In C. E. Weinstein, E. T. Goetz, & P. A. Alexander (Eds.) *Learning and study strategies* (pp. 63–76). San Diego: Academic Press. https://doi.org/10.1016/B978-0-12-742460-6.50011-6. http://www.sciencedirect.com/science/article/pii/B9780127424606500116.

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

Gauch, B., & Biswas, G. (2016). Behavior changes across time and between populations in open-ended learning environments. In *International Conference on Intelligent Tutoring Systems* (pp. 187–196). Springer.

Gillies, R. M. (2003). The behaviors, interactions, and perceptions of junior high school students during small-group learning. *Journal of Educational Psychology*, *95*(1), 137.

Glickman, M. E., Rao, S. R., & 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.

Greene, J. A., & Azevedo, R. (2010). The measurement of learners' self-regulated cognitive and metacognitive processes while using computer-based learning environments. *Educational Psychologist*, *45*(4), 203–209.

Grosch, J., & Emmelmann, H. (1990). A tool box for compiler construction. In *International Workshop on Compiler Construction* (pp. 106–116). Springer.

Grover, S., Bienkowski, M., Niekrasz, J., & Hauswirth, M. (2016). Assessing problem-solving process at scale. In *Proceedings of the third (2016) ACM conference on learning@ scale* (pp. 245–248).

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

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

Hannafin, M. J., Hill, J. R., Land, S. M., & Lee, E. (2014). *Student-centered, open learning environments: research, theory, and practice*, (pp. 641–651). New York: Springer New York. https://doi.org/10.1007/978-1-4614-3185-5_51.

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

Hestenes, D., Wells, M., & Swackhamer, G. (1992). Force concept inventory. *The Physics Teacher*, *30*(3), 141–158.

Hmelo-Silver, C. E. (2004). Problem-based learning: What and how do students learn? *Educational Psychology Review*, *16*(3), 235–266.

Hopcroft, J. E., Motwani, R., & Ullman, J.D. (2001). Introduction to automata theory, languages, and computation. *Acm Sigact News*, *32*(1), 60–65.

Hutchins, N. M., Biswas, G., Maróti, M., Lédeczi, Á., Grover, S., Wolf, R., Blair, K. P., Chin, D., Conlin, L., Basu, S., & et al. (2020a). C2stem: a system for synergistic learning of physics and computational thinking. *Journal of Science Education and Technology*, *29*(1), 83–100.

Hutchins, N. M., Biswas, G., Zhang, N., Snyder, C., Lédeczi, Á., & Maróti, M. (2020b). Domain-specific modeling languages in computer-based learning environments: a systematic approach to support science learning through computational modeling. *International Journal of Artificial Intelligence in Education*, *30*(4), 537–580.

Jacobson, M. J., & 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.

Juhaňák, L., Zounek, J., & Rohlíková, L. (2019). Using process mining to analyze students' quiz-taking behavior patterns in a learning management system. *Computers in Human Behavior*, *92*, 496–506.

Käser, T., & Schwartz, D. L. (2020). Modeling and analyzing inquiry strategies in open-ended learning environments. *International Journal of Artificial Intelligence in Education*, *30*(3), 504–535.

Kinnebrew, J. S., Loretz, K. M., & 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., & 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., & 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. https://doi.org/10.1109/TLT.2015.2513387.

Kovanović, V., Gašević, D., Joksimović, S., Hatala, M., & Adesope, O. (2015). Analytics of communities of inquiry: Effects of learning technology use on cognitive presence in asynchronous online discussions. *The Internet and Higher Education*, *27*, 74–89.

Land, S. M. (2000). Cognitive requirements for learning with open-ended learning environments. *Educational Technology Research and Development*, *48*(3), 61–78. https://doi.org/10.1007/BF02319858.

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

Liu, R., Patel, R., & Koedinger, K.R. (2016). Modeling common misconceptions in learning process data. In *Proceedings of the Sixth International Conference on Learning Analytics & Knowledge* (pp. 369–377).

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

Maldonado-Mahauad, J., Pérez-sanagustín, M., Kizilcec, R. F., Morales, N., & Munoz-Gama, J. (2018). Mining theory-based patterns from big data: Identifying self-regulated learning strategies in massive open online courses. *Computers in Human Behavior*, *80*, 179–196.

Matcha, W., Gašević, D., Jovanović, J., Pardo, A., Maldonado-Mahauad, J., & Pérez-Sanagustín, M. (2019). Detection of learning strategies: A comparison of process, sequence and network analytic approaches. In *European conference on technology enhanced learning* (pp. 525–540). Springer.

Mayer, R.E. (1988). 2 - learning strategies: An overview. In C. E. Weinstein, E. T. Goetz, & P. A. Alexander (Eds.) *Learning and study strategies* (pp. 11–22). San Diego: Academic Press. https://doi.org/10.1016/B978-0-12-742460-6.50008-6. http://www.sciencedirect.com/science/article/pii/B9780127424606500086.

McElhaney, K., Zhang, N., Basu, S., McBride, E., Biswas, G., & Chiu, J. (2020). Using computational modeling to integrate science and engineering curricular activities. In M. Gresalfi, & I. S. Horn (Eds.) *The interdisciplinarity of the learning sciences, 14th international conference of the learning sciences (ICLS) 2020*, Vol. 3.

Mendez, G. R., du Boulay, B., & 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* (pp. 459–466).

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

Munshi, A., Rajendran, R., Ocumpaugh, J., Biswas, G., Baker, R. S., & 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* (pp. 131–138). ACM.

Neamtiu, I., Foster, J. S., & Hicks, M. (2005). Understanding source code evolution using abstract syntax tree matching. In *Proceedings of the 2005 international workshop on Mining software repositories* (pp. 1–5).

Newman, R. S. (1990). Children's help-seeking in the classroom: The role of motivational factors and attitudes. *Journal of Educational Psychology*, *82*(1), 71.

Nisbet, J., & Shucksmith, J. (2017). Learning strategies, Routledge, London.

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.

Obergriesser, S., & Stoeger, H. (2020). Students' emotions of enjoyment and boredom and their use of cognitive learning strategies–how do they affect one another? *Learning and Instruction*, *66*, 101285.

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

Panadero, E., & 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., & 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.

Parkinson, B., & Totterdell, P. (1999). Classifying affect-regulation strategies. *Cognition & Emotion*, *13*(3), 277–303.

Pelánek, R. (2017). Bayesian knowledge tracing, logistic models, and beyond: an overview of learner modeling techniques. *User Modeling and User-Adapted Interaction*, *27*(3), 313–350.

Piech, C., Spencer, J., Huang, J., Ganguli, S., Sahami, M., Guibas, L., & Sohl-Dickstein, J. (2015). Deep knowledge tracing. arXiv:1506.05908.

Pressley, M., Goodchild, F., Fleet, J., Zajchowski, R., & Evans, E.D. (1989). The challenges of classroom strategy instruction, (Vol. 89 pp. 301–342). http://www.jstor.org/stable/1001806.

Rabinovich, M., Stern, M., & Klein, D. (2017). Abstract syntax networks for code generation and semantic parsing. arXiv:1704.07535.

Roscoe, R. D., Segedy, J. R., Sulcer, B., Jeong, H., & Biswas, G. (2013). Shallow strategy development in a teachable agent environment designed to support self-regulated learning. *Computers & Education*, *62*, 286–297.

Roscoe, R. D., Allen, L. K., & McNamara, D.S. (2019). Contrasting writing practice formats in a writing strategy tutoring system. *Journal of Educational Computing Research*, *57*(3), 723–754.

Rosenthal, R., Cooper, H., & Hedges, L. (1994). Parametric measures of effect size. *The Handbook of Research Synthesis*, *621*, 231–244.

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

Schraagen, J. M., Chipman, S. F., & Shalin, V.L. (2000). Cognitive task analysis. Psychology Press.
Schraw, G., Crippen, K. J., & 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.
Schunk, D., & Greene, J. (2018). Handbook of self-regulation of learning and performance. London: Routledge. https://doi.org/10.4324/9781315697048.
Segedy, J. R., Kinnebrew, J. S., & 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* (pp. 449–458). Springer.
Segedy, J. R., Kinnebrew, J. S., & 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., Kinnebrew, J. S., Basu, S., Biswas, G., & 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., & Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review*, *22*, 142–158.
Smit, K., de Brabander, C. J., Boekaerts, M., & 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.
Snyder, C., Biswas, G., Emara, M., Grover, S., & Conlin, L. (2019). Analyzing students' synergistic learning processes in physics and ct by collaborative discourse analysis. Computer-supported collaborative learning.
Taub, M., Mudrick, N. V., Azevedo, R., Millar, G. C., Rowe, J., & Lester, J. (2017). Using multi-channel data with multi-level modeling to assess in-game performance during gameplay with crystal island. *Computers in Human Behavior*, *76*, 641–655.
Taub, M., Mudrick, N. V., Rajendran, R., Dong, Y., Biswas, G., & Azevedo, R. (2018). How are students' emotions associated with the accuracy of their note taking and summarizing during learning with itss?. In R. Nkambou, R. Azevedo, & J. Vassileva (Eds.) *Intelligent tutoring systems* (pp. 233–242). Cham: Springer International Publishing.
Taub, M., Azevedo, R., Rajendran, R., Cloude, E. B., Biswas, G., & Price, M.J. (2019). How are students' emotions related to the accuracy of cognitive and metacognitive processes during learning with an intelligent tutoring system? Learning and Instruction.
Tsai, M. J., & Tsai, C. C. (2003). Information searching strategies in web-based science learning: The role of internet self-efficacy. *Innovations in Education and Teaching International*, *40*(1), 43–50.
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. In *Handbook of Strategies and Strategic Processing* (p. 118).
de Vries, E., Demetriadis, S., & Ainsworth, S. (2009). *External representations for learning*, (pp. 137–153). Berlin: Springer.
Vygotsky, L. S. (1978). Mind in society: The development of higher mental process.
Wang, F., & Hannafin, M. J. (2005). Design-based research and technology-enhanced learning environments. *Educational Technology Research and Development*, *53*(4), 5–23.
Weinstein, C., & Meyer, D. (1994). Learning strategies, teaching and testing. *The International Encyclopedia of Education*, *2*, 3335–3340.
Weinstein, C. E., Acee, T. W., & Jung, J. (2011). Self-regulation and learning strategies. *New Directions for Teaching and Learning*, *2011*(126), 45–53.
Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology*, *25*(1), 127–147.
Whitelock-Wainwright, A., Laan, N., Wen, D., & Gašević, D. (2020). Exploring student information problem solving behaviour using fine-grained concept map and search tool data. *Computers & Education*, *145*, 103731.
Wilensky, U., & 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., & 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. 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., & Hadwin, A. F. (1998). Studying as self-regulated learning. *Metacognition in Educational Theory and Practice*, *93*, 27–30.

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

Winne, P. H., Jamieson-Noel, D., & 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. New York: Macmillan.

Xu, S., & Chee, Y. S. (2003). Transformation-based diagnosis of student programs for programming tutoring systems. *IEEE Transactions on Software Engineering*, *29*(4), 360–384.

Yett, B., Snyder, C., Zhang, N., Hutchins, N. M., Mishra, S., & Biswas, G. (2020). Using log and discourse analysis to improve understanding of collaborative programming. In *Proceedings of the International Conference on Computers in Education (ICCE)*.

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., & Hidalgo-Pontet, Y. (2014). Evaluation of a meta-tutor for constructing models of dynamic systems. *Computers & Education*, *75*, 196–217.

Zhang, N. (2020). Supporting the integrated learning of science, engineering and computational thinking in an open-ended learning environment. PhD thesis, Vanderbilt University.

Zhang, N., & Biswas, G. (2018). Understanding students' problem-solving strategies in a synergistic learning-by-modeling environment. In *International conference on artificial intelligence in education* (pp. 405–410). Springer.

Zhang, N., Biswas, G., & Dong, Y. (2017). Characterizing students' learning behaviors using unsupervised learning methods. In *International conference on artificial intelligence in education* (pp. 430–441). Springer.

Zhang, N., Biswas, G., McElhaney, K. W., Basu, S., McBride, E., & Chiu, J.L. (2020). Studying the interactions between science, engineering, and computational thinking in a learning-by-modeling environment. In *International conference on artificial intelligence in education* (pp. 598–609). Springer.

Zhang, Y., Jin, R., & Zhou, Z.H. (2010). Understanding bag-of-words model: a statistical framework. *International Journal of Machine Learning and Cybernetics*, *1*(1-4), 43–52.

Zimmerman, B. J. (2000). Chapter 2 - attaining self-regulation: A social cognitive perspective. In M. Boekaerts, P. R. Pintrich, & M. Zeidner (Eds.) *Handbook of self-regulation* (pp. 13–39). San Diego: Academic Press. 10.1016/B978-012109890-2/50031-7. http://www.sciencedirect.com/science/article/pii/B9780121098902500317.