Highlights

Using Collaborative Interactivity Metrics to analyze students' Problem-Solving Behaviors during STEM+C Computational Modeling Tasks

- The impact of synergy, turn-taking and equity on computational modeling differs at the overall task level compared to segment-level performance.
- We leverage context-specific segmentation and large language model summarizations of student discourse and actions to understand segment-level collaborative problem-solving.
- High- and Low-performing groups' planning, enacting and reflecting behaviors during segments of collaborative problem-solving differ.

Using Collaborative Interactivity Metrics to analyze students' Problem-Solving Behaviors during STEM+C Computational Modeling Tasks*

ARTICLE INFO

Keywords:
Collaborative problem solving
Computational Modeling
Synergistic Learning
Multimodal Learning Analytics

ABSTRACT

Recently there has been increased development of curriculum and tools that integrate computing (C) into Science, Technology, Engineering and Math (STEM) learning environments. These environments serve as a catalyst for authentic collaborative problem-solving (CPS) and help students synergistically learn STEM+C content. In this work, we analyzed students' collaborative problem-solving behaviors as they worked in pairs to construct computational models in kinematics. We leveraged social measures, such as equity and turn-taking, along with a domain-specific measure that quantifies the synergistic interleaving of science and computing concepts in the students' dialogue to gain a deeper understanding of the relationship between students' collaborative behaviors and their ability to complete a STEM+C computational modeling task. Our results extend past findings identifying the importance of synergistic dialogue and suggest that while equitable discourse is important for overall task success, fluctuations in equity and turn-taking at the segment level may not have an impact on segment-level task performance. To better understand students' segment-level behaviors, we identified and characterized groups' planning, enacting, and reflection behaviors along with monitoring processes they employed to check their progress as they constructed their models. Leveraging Markov Chain (MC) analysis, we identified differences in high- and low-performing groups' transitions between these phases of students' activities. We then compared the synergistic, turn-taking, and equity measures for these groups for each one of the MC model states to gain a deeper understanding of how these collaboration behaviors relate to their computational modeling performance. We believe that characterizing differences in collaborative problem-solving behaviors allows us to gain a better understanding of the difficulties students face as they work on their computational modeling tasks.

1. Introduction

Recently there has been an increased focus on deploying technology-enhanced learning environments in secondary school classrooms to support problem-based learning (PBL) in Science, Technology, Engineering, and Mathematics (STEM) domains (Asghar et al., 2012; Jeong et al., 2019). In our work, we leverage the connections between science and computing (C) (NRC, 2012) to develop STEM+C curricula that include authentic and complex computational modeling of scientific processes and related problem-solving tasks. While researchers have demonstrated the effectiveness of these STEM+C learning environments in supporting learning across multiple domains (e.g., Sengupta et al., 2013; Weintrop et al., 2016, Anonymized), they have also found that these environments introduce additional complexity that exacerbates students' difficulties in constructing and integrating knowledge during their model building tasks (Basu et al., 2016; Chi, 2008). Our approach to alleviating these concerns is to get students to collaborate so that they can work together to explore, develop ideas, and construct and evaluate solutions for these complex, computational modeling tasks. Collaborative Problem Solving (CPS) approaches have been shown to benefit student learning (Beers et al., 2005; Sears and Reagin, 2013). Previous research has demonstrated that students working together often develop shared knowledge, which makes it easier for them to solve problems (Dillenbourg, 1999; Roschelle and Teasley, 1995).

Research has also shown that open-ended, technology-enhanced problem-solving environments provide a shared situational context can act as a catalyst for fostering authentic collaborative problem-solving (Anonymized). In this paper, we build on previous work and study how a collaborative open-ended, technology-enhanced learning environment (Anonymized) promotes knowledge development, knowledge organization, and problem-solving skills in STEM+C domains. In more detail, we study students' collaboration behaviors in the ABC system (Anonymized) as

*This work is supported by the National Science Foundation under award [Anonymized]. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

ORCID(s):

they engage in inquiry, model construction, and model debugging activities while working on computational problems in kinematics. Our analysis of student learning and collaborative behaviors leverages the planning, enactment, and reflection framework adapted from self-regulated learning (SRL) theory (Schunk and Zimmerman, 1998). In this paper, we apply this framework to better understand students' socially shared regulation of learning (SSRL) processes. We believe that this analysis performed through the lens of multimodal data, i.e., student discourse and log file data, will help us better understand students' productive and unproductive behaviors and the difficulties they face from their dialogue and their problem-solving behaviors in authentic STEM+C PBL contexts.

More specifically, we will study the social aspects of students' collaborative dialog using *turn-taking* and *equity measures* and the *synergistic* nature of their domain-specific conversations. Studying the synergistic nature of students' conversations helps us understand how they develop and combine their science and computational knowledge to support their model-building and debugging activities (Anonymized). Using this exploratory analysis framework, we address the following research questions:

- *RQ1: How well do interaction process measures, such as equity and turn-taking, and the synergistic dialog measure relate to students' model-building performance in kinematics?* and
- *RQ2: When analyzing students' segment-by-segment activities, how do students' collaborative problem-solving behaviors relate to their ability to construct computational models in kinematics?*

We answer RQ1 by performing correlation analyses of group interactivity and synergistic dialog with their overall model scores. We answer RQ2 by expanding our analysis of the interactive process metrics to individual time segments of students' collaborative problem-solving behaviors. Our method for segmentation analyzes students' model-building and debugging activities derived from the collected log data. We compare task and segment-level performance of high- and low-performing groups (using a median split of their final model scores). We then apply Markov Chain analysis (Craig and Sendi, 2002) to compare the differences in the high probability transitions between the high- and low-performing students while planning, enacting, and reflecting during model building and debugging tasks. Our analysis uses manual coding of the large language model (LLM)-generated summaries of group discourse and the actions that students perform in each segment. Overall, these results will help us characterize collaborative problem-solving behaviors and how these relate to the students' abilities to construct computational models.

The rest of this paper is organized as follows. Section 2 provides a background review of research in collaborative learning and outlines how our work extends this research in STEM+C domains. Section 3 discusses our analysis methods. This includes a discussion of our ABC environment and curriculum and the research study we conducted in a high school STEM classroom. Section 4 presents the results of our analyses and our answers to RQ1 and RQ2. Section 5 presents the conclusions of this paper and directions for future work.

2. Background

Collaboration is an important component of learning that supports deeper thinking and the development of advanced problem-solving skills (NRC, 2012). Roschelle and Teasley defined collaboration as *"a coordinated, synchronous activity that is a result of a continuous attempt to construct and maintain a shared conception of a problem"* (Roschelle and Teasley, 1995, p. 70). Communication among group members is important for successful collaborative learning (Rummel et al., 2009). Research has shown the importance of developing a *shared understanding* for successful task completion (Larkin, 2006). In other words, successful collaboration requires active contributions and coordination among members of the group combined with effective social interactions to support *knowledge co-construction* and the application of the constructed domain knowledge to solve problems (OECD, 2015). These interaction skills encompass making and encouraging contributions of ideas, translating them into problem-solving steps, monitoring and reflecting on progress, and providing constructive feedback using argumentation and explanation (Garrison and Akyol, 2013; Grau and Whitebread, 2012). Specific to this work, collaboration is also an integral part of STEM learning and practice. In acknowledging its importance, the Next Generation Science Standards (NGSS, 2013) list collaboration-related practices (such as argumentation and communicating information) as key to science learning.

Simultaneously, the NGSS acknowledges computational thinking as a key science practice in recognition of the growing connections between science and computing (Grover and Pea, 2013; Wing, 2006). Computing in science learning (henceforth referred to as "STEM+C") has been actualized through inquiry tasks and computational modeling (e.g., Hambrusch et al., 2009). There has been substantial research on the benefits of learning science through

computational modeling (diSessa, 2001; Sengupta et al., 2013; Sherin et al., 1993). However, studies have also documented the challenges students face in such settings, including translating science disciplinary knowledge and mathematical relationships to computational forms for model building (Basu et al., 2016). For this research, we leverage CPS, taking advantage of the in-the-moment problem-solving discourse between students to enrich our understanding of STEM+C PBL processes.

Measuring students' collaboration in integrated science and CT curricula requires tracking the concepts and practices they apply across the two domains during their problem-solving tasks (Sengupta et al., 2013). While research has examined the multi-dimensional nature of collaboration for university students working on complex tasks (e.g., Nasir et al., 2021; Järvelä et al., 2020), more research is needed to gain a deeper understanding of how collaborative processes impact learning for K-12 students.

One approach to better understanding students' CPS processes is to categorize their activities into goal-setting and planning, enacting, and reflection behaviors along with monitoring processes they may apply to check their progress during enacting (Winne et al., 2002). Research has shown the importance of planning as students must establish goals associated with accomplishing their tasks, possibly decompose goals into a set of more manageable sub-goals, generate plans to accomplish these goals, and then identify potential strategies for enacting these plans (Eilam and Aharon, 2003). This framework has primarily been applied to study individual students' self-regulated learning processes (Zimmerman and Moylan, 2009). However, when applied in the context of CPS, there is another dimension added to goal-setting and planning behaviors, as students must form a *shared understanding* of their goals and plans and come to a *consensus* on the strategies they will use to facilitate the execution of their plans. Developing and enacting plans is further complicated in STEM+C problem contexts, as students have to navigate their differing knowledge backgrounds and share responsibilities in elaborating on their plans and strategies to enact appropriate problem-solving actions.

Reflection is also a necessary component in effective problem-solving as prior work has highlighted the role of learning cycles and students' adaptation across learning cycles as they reflect on their learning processes (Raković et al., 2022). In addition, students are known to monitor their progress when working on learning and problem-solving tasks (Schwartz et al., 2009). This is especially important in computation modeling because students have to combine their construction and debugging strategies to succeed in their modeling tasks (Anonymized). When students collaborate, they may use their shared understanding of the problem to monitor and debug their models and reflection activities to pause and evaluate their evolving solutions (Kalina and Powell, 2009; Stahl and Hesse, 2009).

Analyzing students' collaborative problem-solving behaviors is further complicated by the need to align and interweave multiple data modalities, such as students' conversations and their activity data collected in log files (Wise et al., 2021). Research on multimodal learning analytics (MMLA) has been focusing on these efforts (Blikstein and Worsley, 2016) and recent calls for advanced understanding of how to leverage MMLA for collaboration analysis have highlighted the need for improved methods that support actionable analysis (Wise et al., 2021). For example, multimodal analysis often necessitates segmenting the data into analyzable and actionable chunks. In the past, researchers have predominantly leveraged learning-adjacent data at set time intervals, such as every 30 seconds. Such arbitrary choices can have a significant impact on the analysis of students' learning behaviors, particularly because they do not leverage specific educational contexts (Knight et al., 2017). In order to truly harness the benefits of collaborative learning and promote productive collaboration to support STEM+C learning, we adopt and develop a contextualized time segmentation approach to analyze students' model-building and debugging activities.

In summary, using an exploratory multimodal approach to analyze students' activities in STEM+C computational modeling tasks, we hope to extend previous research on collaborative learning by using (1) context-targeted segmentation methods to better understand students' collaborative problem-solving behaviors; and (2) combining interactivity and domain-specific metrics to gain a deeper understanding of K-12 students collaborative learning behaviors.

3. Methods

This section describes our STEM+C learning environment, measures for studying collaborative interactivity, the research study we conducted in a high school classroom, and our analysis methods.

3.1. ABC Learning Environment

Our block-based programming environment, ABC, illustrated in Figure 1, helps students learn their science and computing concepts and practices (Anonymized). The environment provides students with a set of domain-specific modeling (physics, in this case) and additional computational blocks that they can drag and drop onto the script area to

build their computational model (Anyonmized). The model can be simulated to observe the behavior of the object(s) on the stage.

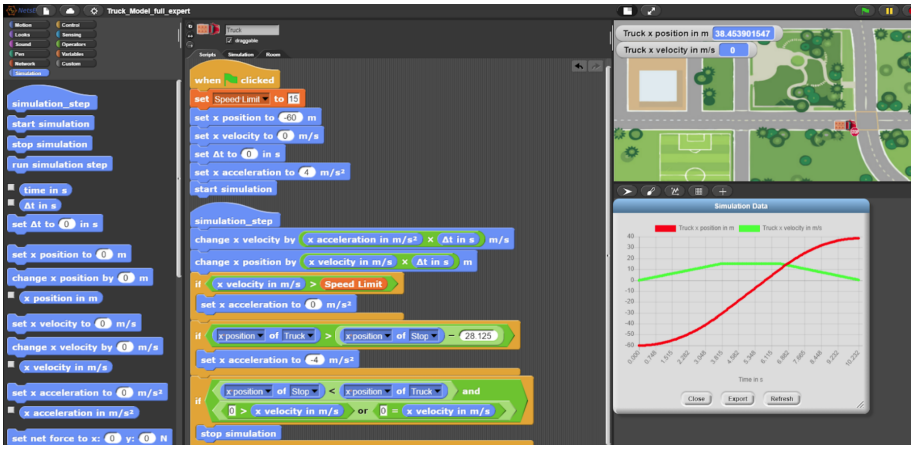

Figure 1: ABC environment

Students working in the ABC system can create partial or complete models and then simulate these models to observe how objects move and how the associated variables change over time. Students typically analyze and debug their evolving models by assessing the motion of the truck object on the stage. In order to support the building and assessment of these complex computational models, the environment provides resources that help students to evaluate and debug their models. Along with animation and variable inspection functions that are displayed on the stage, students have access to graphing and table tools, where chosen variable values are updated dynamically at each simulation step during a simulation run. For example, when variables, such as x-position and x-velocity are selected for display, the simulation run generates position-time and velocity-time graphs, as seen in Figure 1. Students can also use the table tool that is updated with the current x-position and x-velocity of the object at each time step. The graphs and tables help interpret the motion variables to the relevant physics concepts and laws, e.g., the relationship between velocity and acceleration, and make predictions about object behaviors, e.g., how a vehicle speeds up, cruises, and slows down.

Students can also use these features to assess and reflect on the correctness of their models. From a computing perspective, these tools allow for applications of data analysis and debugging that are known to be key computing and model-building practices (Weintrop et al., 2016; Grover et al., 2018). ABC also facilitates collaborative problem-solving as students leverage these tools to co-construct knowledge (Anonymized).

In this study, students used the physics modeling blocks to simulate the motion of the truck that started from rest, sped up to a speed limit, cruised for some time, and then slowed down to come to a stop at a *STOP* sign. Students initialized the relevant physics variables and then used appropriate blocks to update the position and velocity of the truck according to kinematic laws. They used additional computational blocks to create conditional statements that modeled the behavior changes of the truck (Anonymized).

3.2. Research Study and Participants

Our research team conducted a two-month-long study, working with 14-15-year-old 10^{th} grade high school students in a STEM program hosted by a university in the Southeastern United States. The students' backgrounds in computing varied. Some students had completed a high school programming class, while others had no formal experience in programming. None of the students had taken a high school physics course, but some had been introduced to basic kinematics in introductory science classes.

For the study, students were divided into 12 groups (dyads). Each student dyad worked together on a single laptop with a shared mouse and keyboard. They worked on our kinematics curriculum for two hours each week over a period of eight weeks. The data reported in this paper is from their work on a one-dimensional accelerated motion challenge task. The one-dimensional accelerated motion was one of three kinematics units that they worked on.

Our data collection procedure was approved by our university Institutional Review Board. This included collecting summative pre- and post-test assessment data, logged actions in the ABC environment, the final computational models

constructed by each group, and video and audio data using laptop webcams and OBS software. Student actions were recorded in log files with timestamps. Student conversations were automatically transcribed using Otter.ai™, which produced diarized transcripts. Two research team members then edited these transcripts for clarity and accuracy.

We excluded data from one of the groups in our analyses because a student in that group did not consent to our data collection procedure. Two other groups were excluded from our analyses because of problems with audio data collection during the study.

3.3. Data Analysis Methods

Our data analysis procedures included three key components: (1) Measures to quantify students' collaborative interactions as they worked on their collaborative learning tasks; (2) context-targeted segmentation of time-aligned multimodal data; and (3) LLM-generated summaries from the conversations extracted from each time-aligned segment. We discuss our analysis methods in greater detail below and describe how these methods are combined to better understand students' collaborative problem-solving behaviors.

3.3.1. Measuring Collaborative Interactions During Computational Modeling in Science

We measure collaborative interactions along three dimensions: (1) *social*, where students interact with their partners to generate a common understanding of the problem-solving task (Jeong and Chi, 2007); (2) *domain-specific*, where students work together to acquire and combine their science and computing knowledge (Anonymized) ; and (3) *knowledge application performance*, where we evaluate students' STEM+C learning and model bulding progress.

Social Measures of Collaboration

We adopt two interactivity measures that have been used in past research to study the *social* aspects of students' collaborative problem-solving:

1. *equity* (EQU) in students' dialog, i.e., the balance in the amount that each student contributes to the conversation; and
2. *turn-taking* (TT), i.e., how much do students respond to each other's statements and questions as they work together?

Equity measures 'symmetry' in students' conversations, and helps them to negotiate differing perspectives and achieve common understanding (Meier et al., 2007). For students working in pairs, the equity measure was calculated by evaluating the number of utterances made by each student using the following formula:

$$1 - abs(\frac{\#utterances_{S1} - \#utterances_{S2}}{max(\#utterances_{S1}, \#utterances_{S2})}), \tag{1}$$

where $S1$ and $S2$ represent the two students who worked together. The computed value is in the range $[0, 1]$, where a value closer to 0 indicates more inequity and a value closer to 1 indicates greater equity in the conversations between the students.

Turn-taking promotes back-and-forth conversations and helps students to build a common understanding by question posing, explanation, and argumentation (Jeong and Chi, 2007; Soller, 2001). The turn-taking measure computed the number of times the students switched from one speaker to the other. For example, if, during a segment, the utterances had the following pattern: S1, S1, S2, S1, S2, then there would be three switches, while a speaking pattern of S1, S1, S1, S2, S2 has only one switch. The turn-taking measure for each segment was computed as:

$$\frac{\#utterance\ switches}{\#utterances - 1}. \tag{2}$$

A value of 1 implies that the students took turns when speaking over an interval in time (i.e., a segment); while a value of 0 implies one student spoke for the entire time and there was no turn-taking.

Domain-Specific Measure

We extend our analyses by measuring the *synergistic content*, i.e., the interleaving of science and computing concepts in the students' dialog (Anonymized). Two researchers hand-coded approximately 20% of the students'

conversation segments based on the physics and computation concepts they discussed and achieved a Cohen's kappa value of 0.83. Using this coding scheme, we calculated a synergistic score for each segment using the following formula:

$$1 - abs(\#utterances_{Computing} - \#utterances_{Physics}). \quad (3)$$

This computed value was then normalized to the range $[0, 1]$, where a value closer to 0 indicated low synergistic discourse (i.e., conversations in this segment were more focused on one domain), whereas a value closer to 1 indicated high synergistic discourse (i.e., conversations in this segment included concepts in both domains).

Knowledge Application Measures Students' overall STEM+C learning was computed by scoring their final models with a pre-defined rubric generated using a systematic evidence-centered design approach (ECD; Mislevy and Haertel, 2006). The rubrics are discussed in (Anonymized). The groups' overall model-building performances (PERF), normalized to a [0,1] score, are listed in column 2 of Table 3 in Section 4.1 below. We used these PERF scores to divide the students into two groups, high performers and low performers, using a median split to define the groups.

We also evaluated their model-building progress by scoring the students' current computational models at the segment level (see segmentation method in Section 3.3.2 below). We hand-coded their model representations on a qualitative scale into the following categories: (1) *Consistent progress*, if at the end of the segment, there were no errors in the model and the students had added to their previous model; (2) *Some progress*, if at the end of the segment, there was at least one less error in the computational model as compared to the model in the previous segment and they may or may not have added to their computational model; (3) *No progress*, if at the end of the segment, the students had added to their computational model but they did not fix errors from the previous segment; and (4) *Backward progress*, if at the end of the segment, the students had added new errors to their model. These qualitative scores were mapped on a quantitative scale of 0-3 for analysis with 0 corresponding to backward progress, 1 corresponding to no progress, 2 corresponding to some progress, and 3 corresponding to consistent progress.

3.3.2. Context-targeted segmentation

Unlike commonly used methods for segmenting multimodal data that use learning-adjacent data segmentation (e.g., arbitrary pre-defined time segments), we developed a novel segmentation method that aligns well with our problem context. We first encoded students' computational model-building actions as abstract syntax trees (ASTs). Traditionally, ASTs are tree-structured representations used by compilers to capture the syntactic structure of computer programs (Grosch and Emmelmann, 1990). Using these ASTs, we classified students' model-building actions (i.e., adding, removing, adjusting, moving, and populating blocks) into the following high-level categories: (1) *initialization* of relevant variables; (2) *variable updating* in the simulation loop to capture the dynamic behavior of the system; (3) *conditional statements* that primarily captured changes in the dynamic behavior; and (4) *variable updating governed by specific conditions* that were linked to the conditional statements. We illustrate the four high-level categories with example code shown in Figure 2.

A segment ends when students' model-building actions switch from one of these four categories to another. For example, if a group first added blocks to create the conditional statement to get the truck into the cruising mode (i.e., *if x velocity > 15 m*; see Figure 2) this would imply a change in context (e.g., from speeding-up to cruising mode). Next, if the students added the block, *set x acceleration to 0 m/s*, the earlier segment classified as an *conditional statement* segment would end, and a new segment, *variable updating governed by specific conditions* would begin.

Segment categorization only uses the log data. However, the log data provides no information on why the students switched context, and what they planned to do after introducing the context switch in their computational model.On the other hand, the conversations the students have just before and during this segment provide us with much more information about students' problem-solving behaviors. For example, assume a group made a plan to enact the cruise mode of travel from the speed-up mode. To implement this, they enacted the conditional statement, and the subsequent conversation implied that they realized they had forgotten to initialize a variable, i.e., set the *x acceleration* to 0 to correctly implement the cruise mode of travel. More generally, this segmentation and categorization helps us align the multimodal data (logs and discourse) into time intervals and then use LLMs to summarize their conversations in each segment. This presents a systematic approach to understanding and interpreting students' behaviors in each segment of their model-building task. In our study, the average length of a segment was one minute and 49 seconds.

We used additional information to classify each extracted model building segment as a *construction* or a *debugging* episode. For construction episodes, students added and filled into blocks in their current model. For debugging

episodes, students reviewed code that they had already created. Debugging episodes included actions like 'run simulation' and/or 'use the data tools'. In addition, any adjustments, removal, additions, and moving of blocks after the model was assessed were also classified as part of debugging episodes. As an example, if a student group started working on a conditional statement that modeled the truck motion changing from cruise motion (constant velocity) to a slow-down motion with the goal of making it stop at the stop sign, all the model-building actions connected to creating the new conditional statement were classified as part of a construction episode because the students were adding new constructs to their model. On the other hand, if the students ran a simulation after creating the slow down to stop motion and made changes to the blocks in that part of their code, then the associated actions were classified as part of a debugging episode since students were making adjustments or changes to an already existing part of their computational model.

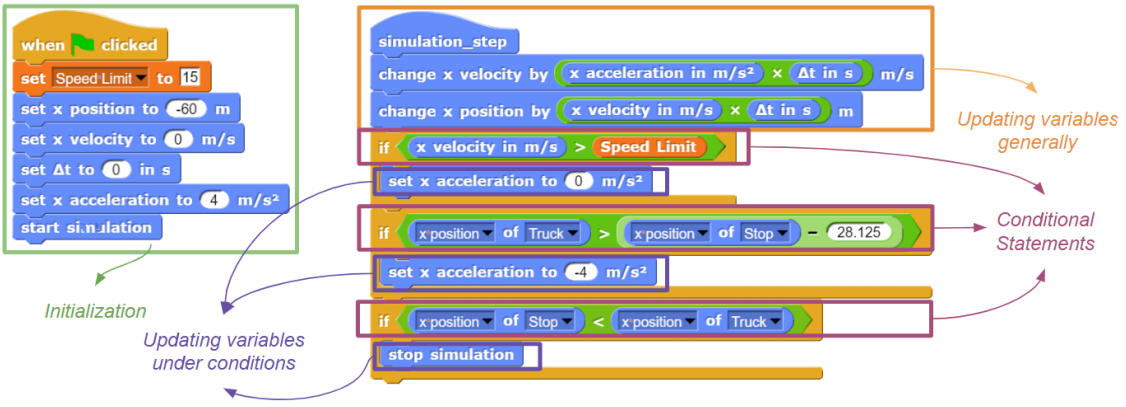

Figure 2: ABC task-specific context used for segmentation

3.3.3. LLM-generated discourse summaries

RQ2 probes deeper into segment-by-segment analysis of students' model construction and their learning behaviors. As discussed earlier, we processed group discourse for each segment generated (see Section 3.3.2). We used GPT-3.5, a transformer-based large language model (LLM) (Vaswani et al., 2017) linked to ChatGPT[1]. Our goal was to use the LLM to summarize students' conversations to help us gain a deeper understanding of their collaborative problem-solving behaviors. For example, the summary could help us analyze how two students planned and enacted the model for the truck slow-down segment. The summaries could also provide insight into their reflection process and how they went through the debugging process when they had errors in this segment of their model.

However, this analysis approach is inherently limited for three primary reasons: (1) the LLM model was pre-trained and it was not possible to use conventional approaches to fine-tune the deep learning model using re-training methods; therefore, summary generation was a "black box" process; (2) we ran into token limitation problems when processing conversation segments; and (3) the lack of knowledge persistence (across segments) limited our ability to capture the temporal evolution in student thinking across multiple segments.

To overcome these limitations to some extent, we developed an initial process that involved two steps: (1) exploring the knowledge retrieval and generation scope of the LLM, and (2) engineering and utilizing existing prompt patterns so that GPT 3.5 could generate reasonably accurate LLM summarizations to answer RQ2. This human-in-loop training framework is outlined in Table 1. The exploration phase targeted two key processes: (1) *Exploration of the LLM Scope* and (2) *Learning Output Testing*. During *Exploration of the LLM Scope* we leveraged our collaboration metrics (discussed in Section 3.3.1) to develop prompts targeting key collaboration constructs. This included LLM-generated summaries about the physics and CT domain knowledge that students discussed to support the computational modeling task. During the human-in-the-loop component, the research team reviewed results, memoed key issues in the summaries generated, and iteratively refined the prompts until identified issues were minimized and researcher consensus on the quality of the summaries was achieved.

[1] https://openai.com/blog/chatgpt

Table 1
Framework for Prompt Generation to Support Context-based Segment Summarizations

Process	Description	Example
Exploration of LLM Scope	Testing preliminary prompt patterns to identify deficiencies in LLM knowledge base and retrieval	An initial prompt was generated for ChatGPT to summarize conversations between students that were working collaboratively in a computer-based learning environment. The input context included information about the learning environment and the domain-specific problem the students had to solve. This process was supported by the *Fact Check List* and *Reflection* (White et al., 2023a) patterns that prompt the LLM to provide the rationale behind its output.
Learning Output Testing	Generating the input knowledge needed for LLM to produce a correct learning task solution (i.e., what knowledge is needed to solve the same problem as students)	Utilizing an iterative process, we developed a combination of inputs that resulted in the LLM generating a correct solution for the problem task. We developed a *Code Generation* pattern that helped the LLM to solve the computational model problem assigned to students. As a result, the LLM learned to summarize the processes that students were utilizing for building that segment model.
Input Semantics Generation	Meta language creation based on the previous two tasks to maximize LLM's understanding of input and ability to process input content	Based on the prompt pattern for Meta Language Creation (White et al., 2023a), we provided ChatGPT with information to better understand the semantics of the input so that it could generate adequate output. As an example, we identified individual student utterances using the label *SPEAKER:DISCOURSE* and prompted ChatGPT to summarize each student's contribution to the conversation.
Output Customization	Leveraging a combination of prompt generation patterns from literature (e.g., White et al., 2023b) to adapt to the requirements of the current research study (i.e., what is the research goal supported by the LLM-generated summary?)	We used an iterative prompt generation process leveraging previously identified prompt patterns supporting output generation to best target our research goal of relating group problem-solving behaviors with their ability to complete the computational modeling task (RQ2). Our experiments produced a prompt that combined the *Meta Language Creation* pattern for input semantics, the *Persona* pattern, and the *Context Manager* patterns based on a manual review of summary outputs by two authors for each prompt iteration.

As an example, during the *Exploration of LLM Scope*, the initial LLM summarization process did not recognize key conceptual knowledge components, such as the *lookahead distance* (i.e., the distance from the stop sign when the truck should start slowing down in order to come to a stop) and how it was calculated. We addressed this more generally, by asking ourselves the question: *Could the LLM generate the solution for the problem task assigned to the students*? This resulted in engineering a prompt pattern we called Code Generation during the *Learning Output Testing Process*. During testing, we identified that the LLM needed the problem description, domain-specific knowledge, and context for generating acceptable solutions to the computational modeling task.

This testing approach supported the engineering phase of our human-in-the-loop approach. Leveraging prior work on prompt engineering (e.g., White et al., 2023b), we focused on two key processes during this phase: (1) *Input Semantics Generation* and (2) *Output Customization*. For the Input Semantics Generation, we focused on *Meta Language Creation* based on the previous two tasks to maximize the LLM's understanding of the input (task, discourse). For Output Customization, we focused on aligning the LLM output to match the requirements for answering RQ2. For example, in our *lookahead distance example*, the *Input Semantics Generation* process involved the refinement of the meta language to support more accurate computational modeling task outputs. Then we incorporated this Code Generation prompt pattern into the *Output Customization* step to support our research task (e.g., ensuring the LLM was provided details of the problem task needed to generate accurate summaries of each group's problem-solving behaviors). For example, in the case of the *lookahead distance* concept, by including these details, the LLM was able

Table 2
Coding Scheme for LLM-generated summaries

Code	Description	Example: Portion of LLM summary
PLANNING	Students are decomposing the problem and/or discussing steps that need to be taken to construct the simulation	The students are discussing how to create a conditional statement for the truck's motion. They consider using an "if-else" statement to check if the velocity equals 15 m/s, and if it does, the truck should cruise. If not, they discuss the possibility of using nested "if" statements to calculate the distance or time remaining until the stop sign and decelerate accordingly.
ENACTING	Students are discussing the actions they are taking to construct the model	One student asks what value to use for delta t, and another student suggests using 0.1. The first student thanks the second student for the suggestion and confirms the value of 0.1
REFLECTING	Students are reviewing the simulation behavior and/or discussing the implications of their constructed code	The students are discussing an issue with the truck's motion model. They mention a block that did not work and discuss a variable with a value of 0.1, which they believe is causing the truck to move slower and smoother.
PLANNING & ENACTING	Students are PLANNING and ENACTING in the same segment	They are trying to figure out how to make the truck slow down and one student suggests that they need to add a new part to the model to make it slow down. They are using a trial-and-error approach to refine their model and are open to suggestions from each other.
PLANNING & REFLECTING	Students are PLANNING and REFLECTING in the same segment	S7 suggests they should check if the truck slows down when it reaches the speed limit, but S21 disagrees and suggests looking at the graph. S7 then confirms that the truck does maintain constant velocity and suggests including velocity in the conditional statement. S21 agrees and apologizes for the confusion. S7 begins to suggest how the conditional statement should be structured.
ENACTING & MONITORING	Students are ENACTING and MONITORING in the same segment	One student changes the velocity value and another student observes that it is not increasing. The first student asks why and the second student responds that it is just not working and taking a long time. The second student then suggests that maybe it is because the "if" statement may not be working.
ENACTING WITH NO DISCUSSION	Students are performing actions (i.e. ENACTING) but there is no discussion.	There is no conversation to summarize.
ASSISTANCE	Students are receiving help from one of the researchers	They are trying to figure out how to make the truck accelerate until it reaches the speed limit of 15 m/s. The researcher suggests using an if statement and looking at the simulation data to determine the appropriate conditions.
OFF-TOPIC	Students are having an off-topic discussion unrelated to the task.	The conversation is not related to the task of creating a computational model of truck motion.

to accurately identify instances in student conversations in which they were discussing the distance between the trucks' position and stop sign in order to calculate when the truck should begin to start slowing down.

Overall, this two-step process resulted in a prompt with the following components:

- a *context manager pattern* (allowing for control of the context of the LLM's output (White et al., 2023a)) that described the model-building task and incorporated the components such as the Code Generation pattern described in our example;
- a *persona pattern*, described by White et al. as an approach that "gives the LLM a persona or role to play when generating output" (p.4, 2023a), in which we indicated to the LLM that it would play the role of a teacher trying to interpret the students' conversations; and

- a *task-specific pattern* that indicated which of the four task-specific segment types a particular group was working on and what that meant in the model building context (e.g., if the segment was labeled as initialization, the prompt stated: “In this segment, the students are working on assigning initial values to variables, such as position and velocity of the truck.").

The prompt concluded with an input semantics statement that outlined the format of the transcript.

3.3.4. Analyzing Planning, Enacting and Reflecting Behaviors using Markov Chain Analysis

The LLM-generated summaries provided an overview of group problem-solving behaviors during each segment. To extract and analyze these behaviors, we hand-coded the summaries based on prior SRL research (Zimmerman and Moylan, 2009; Winne, 2010; White et al., 2009) described in Section 2, using the coding scheme in Table 2. As these summaries were generated using multi-modal data, the identification of their behaviors was based on students' conversations and actions. In addition, we characterized segments into additional categories: (1) students were receiving help from the researcher; (2) having primarily off-topic discussions; and (3) performing actions but having no discussion at all. The full details of the coding scheme, as well as example LLM-generated summaries, can be seen in Table 2.

In order to fully evaluate students' problem-solving, we considered the temporal nature of their behaviors using Markov Chain analysis (Craig and Sendi, 2002). Using an exploratory analysis approach, we systematically reviewed all high-probability transitions with probabilities ≥ 0.2 (a chosen threshold) and described the differences between high- and low-performing groups to link specific behaviors described in Table 2 to students' ability to construct models. In addition, we describe the differences between these behaviors in terms of our CPS framework using the synergistic, turn-taking, and equity metrics described in section 3.3.1.

4. Results

This section presents our approach to answering the two research questions and discusses the results of our analyses.

4.1. RQ1: How well do interaction process measures, such as equity and turn-taking, and the synergistic dialog measure relate to students' model-building performance?

To answer RQ1, we computed the equity (EQU), turn-taking (TT), and synergistic (SYN) scores for each one of the segments in the students' model construction work for all nine groups. In addition, we computed aggregated scores for the full model-building episode by averaging the values across all segments. Since the number of groups was small, we assumed a non-normal distribution and computed the Spearman rank correlation to evaluate the relationships between the performance measures and the collaborative interaction metrics. We used two performance measures: (1) the final model score (PERF), and (2) the total number of actions performed to build the model (ACT, as a measure of their effort) and computed the correlations between these two measures and the three collaborative interaction measures. We also computed the correlation between these two performance measures and ratio of constructing to debugging actions.

Table 3 summarizes all of the pairwise correlation values and their corresponding p-values. The high-performing groups have more synergistic dialogue (SYN) with values in the range $[0.72, 0.82]$) than the low-performing groups, whose synergistic dialogue measure varies in the range $[0.50, 0.69]$. The exception is group G12, which had SYN $= 0.58$.

When considering the social interactive measures of TT and EQU, the groups had overlapping equity and turn-taking values. Excluding G12's scores, we got overlapping TT scores for the high group (in the interval $[0.39, 0.59]$) and the low group (in the interval $[0.30, 0.48]$). Similarly, EQU for the high group was in the interval $[0.53, 0.92]$ whereas, for the low group, it was in the interval $[0.84, 0.91]$. Though the intervals overlapped, the low performers have a somewhat higher aggregated equity in their conversations.

While the highest number of total actions (374) and the lowest number of total actions (145) were taken by the high-performing groups, G2 and G11, respectively, the low-performing groups performed a higher number of total actions on average, i.e., an average of $275.3(sd = 60.8)$ actions per group versus $234.8(sd = 81.4)$ actions per group.

We answered RQ1 by correlating the performance measures with their aggregated interactivity and behavior measures computed for the entire task. Table 4 summarizes the pairwise correlations along with the p-values. Our results show that synergistic discourse (SYN) was strongly correlated with performance ($\rho = 0.6$), and turn-taking (TT) and the ratio of the construct to debug actions were moderately correlated with performance with ρ values

Table 3
Overall Summary Statistics for the high- (blue) and low-performing (orange) groups. The performance measure(PERF), the interaction measures, synergistic dialog (SYN), Turn-Taking (TT), and Equity (EQU), and behavior measures, the ratio of Construction (CONSTR) to debugging (DEBUG) actions is normalized to [0,1] scores. The total number of actions (ACT) and the number of segments (Num Segs) in students' computational model-building work also appear in the Table.

group	PERF	ACT	Num Segs	SYN	TT	EQU	CONSTR/DEBUG
g3	0.97	228	24	0.81	0.59	0.89	0.30
g2	0.95	374	45	0.72	0.39	0.92	0.74
g11	0.92	145	10	0.79	0.52	0.82	0.84
g9	0.89	165	22	0.82	0.42	0.53	0.82
g12	0.84	262	21	0.58	0.24	0.54	0.82
g8	0.76	172	23	0.62	0.48	0.91	0.70
g4	0.71	293	47	0.50	0.32	0.86	0.34
g7	0.68	309	50	0.63	0.30	0.84	0.50
g5	0.42	327	34	0.69	0.42	0.87	0.30

Table 4
Spearman Correlations with Performance and Effort at Overall Task Level (numbers in parentheses are the p-values)

	SYN	TT	EQU	CONSTR/DEBUG
Performance	0.60	0.52	0.13	0.42
(p-val)	(0.067)	(0.154)	(0.732)	(0.265)
Total Actions	-0.35	-0.54	0.48	-0.57
(p-val)	(0.356)	(0.137)	(0.187)	(0.112)

of 0.52 and 0.42, respectively. This provides evidence that conversations among students with interweaving physics and computation discussions were indicative of good understanding and, therefore, resulted in better model-building performance. These results are in agreement with our earlier work that showed the importance of synergistic dialogue during computational modeling tasks (Anonymized). Similarly, turn-taking, which implies a back-and-forth discussion among the paired students and performing more construction to debugging actions resulted in better model-building performance.

Conversely, the moderate negative correlation between TT and the total number of actions ($\rho = -0.54$) suggests that more turn-taking led to more efficient model building (i.e., fewer actions required to build the model). The moderate negative correlation between the ratio of CONSTR to DEBUG actions to performance ($\rho = -0.57$) shows that those who had more construct than debug actions did better in their overall model construction task. As expected, synergistic dialog was negatively correlated with the total number of actions performed, but the correlation was weak ($\rho = -0.35$).

The equity in conversation measure, EQU, had a weak, positive correlation with PERF ($\rho = 0.13$) but a moderate, positive correlation with Total Actions performed ($\rho = 0.48$), which suggests that an equitable sharing of discourse may not result in less effort (i.e. less model building actions) and success in the task. We investigate this result in greater detail later in the paper. In summary, our analyses demonstrate that synergistic dialog and turn-taking during computational modeling are important collaborative processes that support productive learning and model building.

4.2. RQ2: When analyzing students' segment-by-segment activities, how do students' collaborative problem-solving behaviors relate to their ability to construct computational models in kinematics?

To answer RQ2, we calculated segment-level statistics utilizing the segmentation method described in Section 3.3.2. Table 3 shows the number of segments we derived for each of the 9 groups as they constructed their models. The numbers varied between 10 and 50 segments. We computed the segment-level measures as the average (and standard deviation) of the per-segment (1) performance scores; (2) the number of model-building actions; (3) synergistic (SYN); (4) turn-taking (TT); and (5) equity (EQU) measures to compare the high and low performing groups. These results are presented in Table 5.

We computed the average segment performance score by considering the percentage of segments of each performance type using the coding scheme described in Section 3.3.1. When comparing high- and low-performing

Table 5
Summary Statistics - Segment Level

	Performance	Effort	Interactivity Metrics		
	Progress Score Avg. (SD)	Total Actions Avg. (SD)	SYN Avg. (SD)	TT Avg. (SD)	EQU Avg. (SD)
high	1.35 (0.96)	12.7 (17)	0.74 (0.4)	0.42 (0.3)	0.53 (0.4)
low	1.12 (0.74)	8.0 (12)	0.60 (0.5)	0.36 (0.3)	0.49 (0.4)

Table 6
Correlations with Performance and Effort - Segment Level numbers in parentheses are the p-values

	SYN	TT	EQU
performance (p-val)	0.42 (0.004)	0.05 (0.75)	0.08 (0.59)
Total actions (p-val)	0.21 (0.16)	0.31 (0.04)	0.27 (0.07)

groups, the high performers had more *consistent progress* segments than the low performers (20% to 15%). Similarly, they had more *some progress* segments (16% to 6%), but surprisingly, a higher percentage of *backward progress* segments than the low-performing groups (18% to 15%). Last, the low-performing groups had a higher percentage of *no progress* segments (64%) than the high-performing groups (46%). This breakdown suggests that while the high-performing groups had a higher proportion of productive segments (often correcting multiple errors in one segment), they also had a higher proportion of segments where they showed *backward progress* during model-building. We conjecture that the high-performing groups were more willing to explore different options (sometimes incorrect ones) rather than make no progress like the low-performing groups. The segment-level summary statistics for the performance and interactivity metrics (see Table 5) show that the high-performers had better results than the low-performers.

Like RQ1, we analyzed groups' behaviors by correlating the interactivity metrics with performance scores and effort at the segment level for RQ2. Interestingly, performance and effort were very weakly correlated ($\rho = 0.1$) implying little to no relationship between the number of actions the students performed and their model construction progress at the segment level. Table 6 shows that synergistic discourse was moderately correlated ($\rho = 0.42$) with performance at the segment level, while both TT and EQU had a weak, positive correlation with performance ($\rho = 0.05$ and 0.08, respectively) at the segment level. At the aggregate level, EQU and performance was weakly correlated ($\rho = 0.13$), but the strength of the correlation between TT and performance was moderate ($\rho = 0.52$). Interestingly, the strength of the correlation between the number of actions and TT was moderate and negative at the aggregate level ($\rho = -0.54$) but weak and positive at the segment level ($\rho = 0.31$). This implies turn-taking made students more efficient model builders at the aggregate level, but that relation did not translate to the finer-grained segment level. The correlation between the number of actions and EQU was moderately strong at the aggregate level ($\rho = 0.48$) and weak at the segment level ($\rho = 0.27$). When interpreting these results, we conjecture that more turn-taking in collaborative discourse at the segment level may be linked to exploration and that may lead to varying degrees of progress in model building as discussed above. For example, when students are unsure of how to progress in their model-building task, they may strategically try out alternate plans that each student in the group proposes to determine if one of them leads to progress in their current task.

To explore these problem-solving behaviors further, we conducted a deeper analysis of groups' planning, enacting, and reflecting behaviors at the segment level. We used the Markov Chain (MC) representation to model the behavior differences between the two groups (see Figure 3). To derive the MC model, we used the previously defined hand-coded labels (planning, enacting, reflecting, assistance, off-topic, and enacting with no discussion) generated from the discourse summaries (described in Section 3.3.4) as the states of the model. In the figure, the size of the states is proportional to the relative occurrence of their corresponding process over the entire model-building task (Table 7 lists the numerical values, and we clarify that the values may not exactly sum to 100% due to rounding). An arrow between two states represents a transition between the states, and the number on the arrow captures the transition probability. To reduce clutter and to increase interpretability, we only retain transitions with probability ≥ 0.2. The high-performers spend more time on metacognitive processes (planning and reflection) than the low-performers (29% to 19%), whereas the low-performers spend much more time on cognitive (enactment) processes (53% to 39%). The high-performing groups also spent a little more time in a combined cognitive-metacognitive process, i.e., Enactment and Monitoring.

Table 7
Proportion of Occurrences for each Problem-solving behavior

Behavior	High	Low
Planning	9%	4%
Enacting	16%	24%
Reflecting	8%	6%
Planning and Enacting	20%	17%
Planning and Reflecting	2%	0%
Enacting and Monitoring	15%	10%
Enacting with No Discussion	12%	22%
Assistance	13%	9%
Off-Topic	3%	7%

4.2.1. MC model interpretation

A comparison of the high-performer and low-performer MC models in Figure 3 shows that the high-performing groups have more systematic problem-solving behaviors. For example, one of their chains goes from Planning to Planning and Enacting and then can loop between Enacting and Planning and Enacting or transition to the Enacting and Monitoring state. In other words, they plan part of the model construction task and then start the construction process while continuing to refine their plans. At some point they may gain sufficient confidence to go into a pure model-building phase after which they transition back to Planning and Enacting and then move on to Enacting and Monitoring to check and refine their models. The transition from the Planning and Enacting to the Enacting and Monitoring state very likely represents a transition from model building to model debugging activities. It is interesting to see that at times, after Enacting, the high-performers may seek assistance from the researchers, and our conversations confirm that this was primarily to get the researchers to check and confirm that their model building was on the right track. Alternatively, these students sometimes take a break to Reflect on their model-building progress and then come back to their model-building task. A third sequence that we see among high-performers is that they may take a break (the off-topic state) and then go to the Planning and Reflecting, Enacting, or the Enacting and Monitoring state. Overall, these transitions in the MC model show the high-performers using a systematic approach to their model-building task.

On the other hand, the behavior chains in the MC model for the low-performing group are made up of shorter disjointed sequences of states and state transitions dominated by different Enacting behaviors. Planning, Reflecting, and Enacting and Monitoring (all representative of metacognitive processes) are short and all lead directly to different types of Enactment, i.e. model building by oneself or model building together). Interestingly, like the high-performers, initial solution Planning may lead to Planning and Enacting, but this is an end state, implying the students run into difficulties when model building and are not able to progress further without continous planning. Two other differences between the high- and low-performing groups: (1) low-performing students ask researchers for help on how to progress in their model building, and, therefore, directly transition to the Enactment phase; and (2) low-performers do not see to use their Off-Topic activity productively; they tune out and get disengaged from the model building task.

We also noted that both groups struggled with effective reflection as Reflection phases were few and far between in spite of the fact that both groups were experiencing difficulties in their model-building tasks. Overall, as a response to RQ2, our results suggest that reflecting behavior prior to assistance is important for effective problem-solving. Similarly, effective assistance requires more contextual details that the researchers can use to respond in an effective manner to the students. For example, if the researchers knew that a high-performing group was not reflecting on their model solution before asking for assistance, reflection could be encouraged first. These results identify the need for future work to explore feedback mechanisms that adapt to the process needs of individual students and groups.

4.2.2. Characterizing Behaviors with Interactivity Metrics

In addition to identifying salient problem-solving behaviors, we leveraged the interactivity metrics used to answer RQ1 to better understand the differences between the low- and high-performing groups' Planning, Enacting, Monitoring, and Reflecting behaviors. The high-performing groups had a lower synergistic score during planning than the low-performing groups (0.69 compared to 0.78), suggesting that they may have focused more on one domain at a time while planning, and then switched to a synergistic approach for model building in the enacting phase. In terms of the interactivity metrics, the high-performing groups had a higher average TT score than the low-performing groups

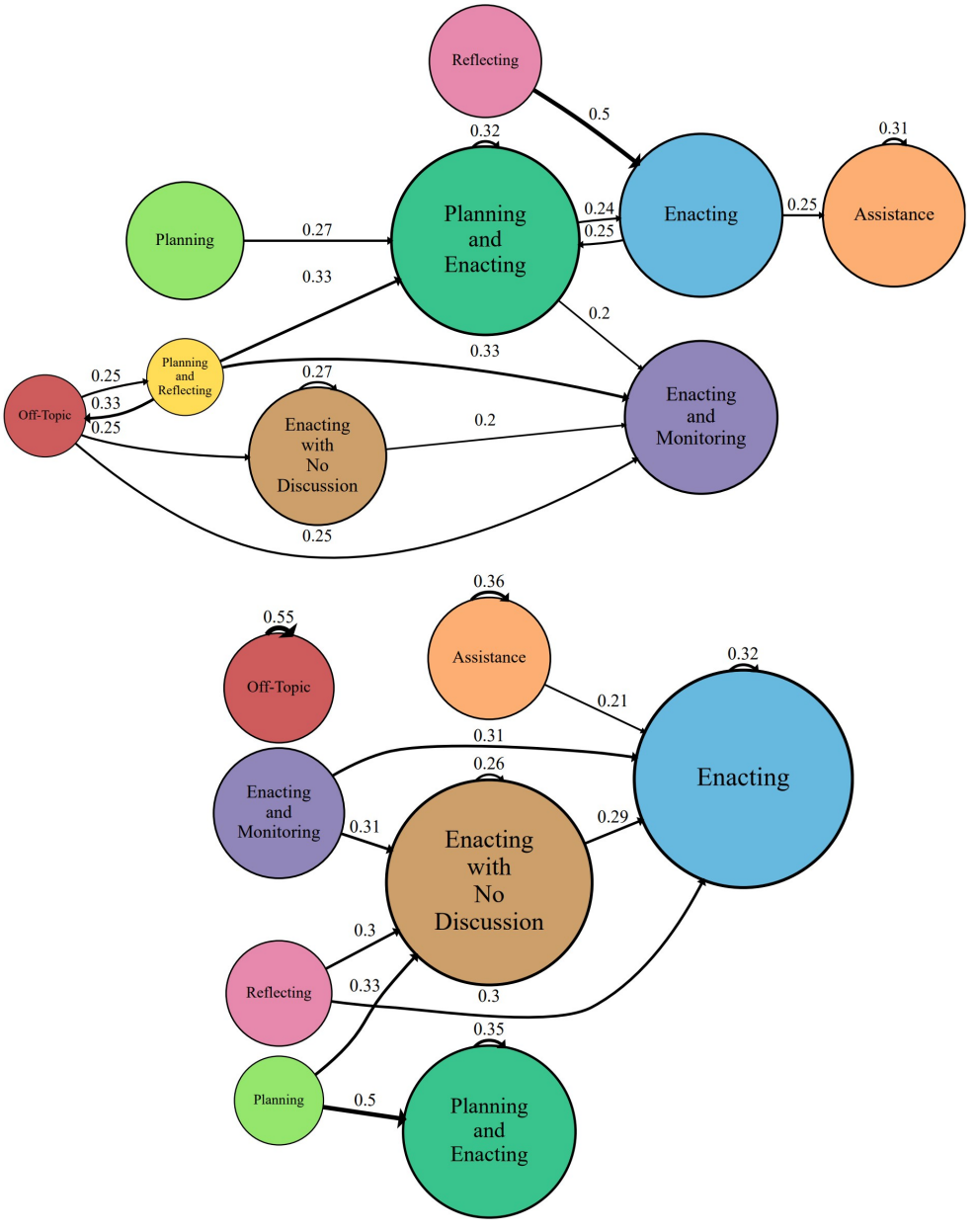

Figure 3: Markov Models for High (top) and Low (bottom) Performing Groups

during planning (0.5 compared to 0.38) but a lower EQU score (0.44 compared to 0.56). From these metrics and the summaries, the evidence points to the low-performing groups' planning often being characterized as the two students laying out their plans with little to no discussion with their partner. In other words, the low-performing groups did not seem to come to a shared understanding, and, therefore, were unable to assist one another during the enactment phase. In contrast, we saw different planning behaviors in the high-performing groups as they more often took the time to come to a consensus, with one student suggesting a plan and then both students contributing to adding to the plan and revising it. For RQ2, these results suggest that the existence of planning behavior alone is not enough for effective model construction, rather it must be combined with effective consensus building, which produces a shared understanding of the plan and approach to tackle a problem (Weinberger and Fischer, 2006).

When considering enacting segments, the high-performing groups had high synergistic scores in cases where they were making progress in their model building ($score = 0.98$) or even when they were making no progress ($score = 0.85$). Their interactivity metrics, i.e., TT ($score = 0.47$ vs 0.35) and EQU ($score = 0.57$ vs 0.48), showed similar trends when looking segments in which they made progress versus those with no progress. This suggests that these groups could have demonstrated more TT and EQU in their discussions in the individual segments to improve their performance. In contrast, the low-performing groups show a reverse trend in that their synergistic and interactivity metrics are lower when they making progress compared to when they are not making progress in the model (SYN = 0.73 and 0.89, respectively; TT = 0.28 and 0.43, respectively; and EQU = 0.45 and 0.53, respectively). This suggests that low-performing groups had one student who took the lead during enacting activities, which may have led to short-term progress at the segment level but negatively impacted their overall task performance. Overall, the results show the importance of high interactivity and synergistic discussions when students are involved in model-building tasks.

The high-performing groups had more synergistic discourse during reflecting (0.65) than the low-performing groups (0.48), suggesting that reflecting on the model after they believed they completed a component before moving to the next segment is a positive behavior but it must also be combined with an ability to reflect on both domains simultaneously. In the Enacting and Monitoring state, both groups had high levels of synergistic dialog as the average synergistic score for all groups was higher in segments where they made progress (0.95 and 0.94, for the high- and low- performing groups, respectively) in comparison to segments where they did not make progress (0.90 and 0.84, respectively). Overall, to answer RQ2, these results suggest that while effective planning may focus on only one domain, effective reflection, and monitoring necessitate interweaving discussions between the two domains.

5. Conclusions and Future Work

The complexities of collaborative problem-solving for computational modeling in science provide a unique opportunity to explore individual and group learning and how interactivity and synergistic dialog affect the learning of science and computing concepts during model-building tasks. In this paper, we examined differences in collaborative problem-solving behaviors and their impact on groups' ability to complete computational modeling tasks in kinematics. We identified the impact of interactivity metrics, such as equity and turn-taking, and the synergistic nature of the collaborative discourse on groups' overall ability to build correct computational models.

We analyzed differences between high- and low-performing groups' planning, enacting, monitoring, and reflecting behaviors using Markov Chain models to study how sequences of these behavior phases affected students' model-building performance. By analyzing the interactivity and synergistic metrics, we were able to characterize how different collaborative behaviors among the groups impacted their model-building performance. All of these analyses were supported by multimodal analyses, where we combined the use of students' conversations and activity logs to segment students' model-building activities and then analyze their collaborative behaviors for each activity segment.

Our analysis of collaborative interactivity and domain-specific processes during problem-solving produced a number of findings and potential for future research. First, our results for RQ1, identified the importance of maintaining synergistic dialog, which extended past findings (e.g., Anonymized,Grover et al., 2019) to demonstrate the importance of leveraging both domains consistently across all model-building task components. Second, while past research has also highlighted the importance of equity and turn-taking during collaborative problem-solving, we identified that as students build their models in segments, fluctuations in equity may not be a critical indicator of student collaboration and task completion efficiency. We believe this highlights the need for future research in better understanding collaborative interactions, especially as we design formative feedback approaches to enhance collaboration.

In addition to better understanding collaborative interactivity components, in answering RQ2, we analyzed problem-solving behaviors centered around planning, enacting, monitoring, and reflecting and identified differences in low- and high-performing groups. By first identifying differences in salient transitions between the phases of activity and then furthering this analysis using students' synergistic, turn-taking, and equity metrics, we were able to better link students' problem-solving behaviors to their model-building performance. We found that high-performing groups better leveraged the benefits of monitoring during problem-solving. In addition, we found that enacting behavior without intervening reflecting and/or planning behaviors may lead to short-term success at the segment level but negatively impacts success in overall model construction. In particular, we found that intermediate reflections before returning to enacting had a positive impact on performance but such reflection must include synergistic discussions that cover science and computational concepts. In addition, we found that planning positively impacts performance,

but is only effective if group members come to a consensus and construct the plan together. We found that continuous planning, with small amounts of enacting behavior, may indicate that the group is simply trying different plans without forming a true shared understanding of how to tackle the task. Our results suggest that when groups go directly from enacting to getting assistance, skipping the reflection step, they may be unable to effectively apply the assistance that is provided. Future work targeting adaptive support for problem-solving will aim to encourage monitoring and reflection behaviors as students are involved in their problem-solving tasks.

We recognize the limitations in our approach. First, this work is limited due to its small sample size. We recognize the limitation of categorizing groups based on high- and low-performance in the primary task. In future work, with increased participant numbers, we aim to leverage additional characteristics of collaborative learning and problem-solving for a more nuanced view of group processes. In addition, we maintained a systematic IRR approach to minimize issues of bias and human error in discourse coding. We foresee great potential in the leveraging of large language models (LLMs), especially considering the benefits they provided to enriching our qualitative analysis approach and computing the interactivity metrics from student discourse. However, this work must also consider the limitations of LLMs, such as hallucinations, token space limits, and nonpersistent memory (described previously) when applying such analysis to examine the temporal changes in collaborative problem-solving.

References

Asghar, A., Ellington, R., Rice, E., Johnson, F., Prime, G.M., 2012. Supporting stem education in secondary science contexts. Interdisciplinary Journal of Problem-Based Learning 6, 4.

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, 13.

Beers, P.J., Boshuizen, H.P.E., Kirschner, P.A., Gijselaers, W.H., 2005. Computer support for knowledge construction in collaborative learning environments. Computers in Human Behavior 21, 623–643.

Blikstein, P., Worsley, M., 2016. Multimodal learning analytics and education data mining: Using computational technologies to measure complex learning tasks. Journal of Learning Analytics 3, 220–238.

Chi, M.T.H., 2008. Three types of conceptual change: Belief revision, mental model transformation, and categorical shift, in: Vosniadou, S. (Ed.), Handbook of research on conceptual change. Erlbaum, Hillsdale, NJ, USA, pp. 61—-82.

Craig, B.A., Sendi, P.P., 2002. Estimation of the transition matrix of a discrete-time markov chain. Health Economics 11, 33–42. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/hec.654, doi:https://doi.org/10.1002/hec.654, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/hec.654.

Dillenbourg, P., 1999. Collaborative learning: Cognitive and computational approaches. advances in learning and instruction series. ERIC.

diSessa, A., 2001. Changing minds: Computers, learning, and literacy. MIT Press, Cambridge MA USA.

Eilam, B., Aharon, I., 2003. Students' planning in the process of self-regulated learning. Contemporary educational psychology 28, 304–334.

Garrison, D.R., Akyol, Z., 2013. Th e community of inquiry th eoretical framework, in: Handbook of distance education. Routledge, pp. 122–138.

Grau, V., Whitebread, D., 2012. Self and social regulation of learning during collaborative activities in the classroom: The interplay of individual and group cognition. Learning and Instruction 22, 401–412.

Grosch, J., Emmelmann, H., 1990. A tool box for compiler construction, in: International Workshop on Compiler Construction, Springer. pp. 106–116.

Grover, S., Basu, S., Schank, P., 2018. What we can learn about student learning from open-ended programming projects in middle school computer science, in: Proceedings of the 49th ACM Technical Symposium on Computer Science Education, Association for Computing Machinery, New York, NY, USA. p. 999–1004. URL: https://doi.org/10.1145/3159450.3159522, doi:10.1145/3159450.3159522.

Grover, S., Hutchins, N., Biswas, G., Snyder, C., Emara, M., 2019. Examining synergistic learning of physics and computational thinking through collaborative problem solving in computational modeling, in: The American Educational Research Association Annual Meeting.

Grover, S., Pea, R., 2013. Computational thinking in k–12: A review of the state of the field. Educational Researcher 42, 38–43. URL: https://doi.org/10.3102/0013189X12463051, doi:10.3102/0013189X12463051, arXiv:https://doi.org/10.3102/0013189X12463051.

Hambrusch, S., Hoffmann, C., Korb, J.T., Haugan, M., Hosking, A.L., 2009. A multidisciplinary approach towards computational thinking for science majors. SIGCSE Bull. 41, 183–187. URL: https://doi.org/10.1145/1539024.1508931, doi:10.1145/1539024.1508931.

Järvelä, S., Gašević, D., Seppänen, T., Pechenizkiy, M., Kirschner, P.A., 2020. Bridging learning sciences, machine learning and affective computing for understanding cognition and affect in collaborative learning. British Journal of Educational Technology 51, 2391–2406.

Jeong, H., Chi, M.T., 2007. Knowledge convergence and collaborative learning. Instructional Science 35, 287–315.

Jeong, H., Hmelo-Silver, C.E., Jo, K., 2019. Ten years of computer-supported collaborative learning: A meta-analysis of cscl in stem education during 2005–2014. Educational research review 28, 100284.

Kalina, C., Powell, K., 2009. Cognitive and social constructivism: Developing tools for an effective classroom. Education 130, 241–250.

Knight, S., Wise, A.F., Chen, B., 2017. Time for change: Why learning analytics needs temporal analysis. Journal of Learning Analytics 4, 7–17.

Larkin, S., 2006. Collaborative group work and individual development of metacognition in the early years. Research in science education 36, 7–27.

Meier, A., Spada, H., Rummel, N., 2007. A rating scheme for assessing the quality of computer-supported collaboration processes. International Journal of Computer-Supported Collaborative Learning 2, 63–86.

Mislevy, R.J., Haertel, G.D., 2006. Implications of evidence-centered design for educational testing. Educational Measurement: Issues and Practice 25, 6–20. URL: https://onlinelibrary.wiley.com/doi/

abs/10.1111/j.1745-3992.2006.00075.x, doi:https://doi.org/10.1111/j.1745-3992.2006.00075.x, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1745-3992.2006.00075.x.

Nasir, J., Kothiyal, A., Bruno, B., Dillenbourg, P., 2021. Many are the ways to learn identifying multi-modal behavioral profiles of collaborative learning in constructivist activities. International Journal of Computer-Supported Collaborative Learning 16, 485–523.

NGSS, 2013. Next Generation Science Standards: For States, By States. The National Academies Press.

NRC, 2012. A framework for K-12 science education: Practices, crosscutting concepts, and core ideas. National Academies Press.

OECD, 2015. PISA 2015 Collaborative problem solving framework, Retrieved from www.oecd.org/pisa/pisaproducts/Draft%20PISA%202015%20.

Raković, M., Bernacki, M.L., Greene, J.A., Plumley, R.D., Hogan, K.A., Gates, K.M., Panter, A.T., 2022. Examining the critical role of evaluation and adaptation in self-regulated learning. Contemporary Educational Psychology 68, 102027.

Roschelle, J., Teasley, S.D., 1995. The construction of shared knowledge in collaborative problem solving, in: Computer supported collaborative learning, Springer. pp. 69–97.

Rummel, N., Spada, H., Hauser, S., 2009. Learning to collaborate while being scripted or by observing a model. International Journal of Computer-Supported Collaborative Learning 4, 69–92.

Schunk, D.H., Zimmerman, B.J., 1998. Self-regulated learning: From teaching to self-reflective practice. Guilford Press.

Schwartz, D., Chase, C., Chin, D., Oppezzo, M., Kwong, H., Okita, S., Biswas, G., Roscoe, R., Jeong, H., Wagster, J., 2009. Interactive metacognition: Monitoring and regulating a teachable agent. Handbook of metacognition in education , 340–359.

Sears, D.A., Reagin, J.M., 2013. Individual versus collaborative problem solving: Divergent outcomes depending on task complexity. Instructional science 41, 1153–1172.

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, 351–380.

Sherin, B., diSessa, A.A., Hammer, D., 1993. Dynaturtle revisited: Learning physics through collaborative design of a computer model. Interactive Learning Environments 3, 91–118. URL: https://doi.org/10.1080/1049482930030201, doi:10.1080/1049482930030201, arXiv:https://doi.org/10.1080/1049482930030201.

Soller, A., 2001. Supporting social interaction in an intelligent collaborative learning system. International journal of artificial intelligence in education 12, 40–62.

Stahl, G., Hesse, F., 2009. Paradigms of shared knowledge. International Journal of Computer-Supported Collaborative Learning 4, 365–369.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I., 2017. Attention is all you need. Advances in neural information processing systems 30.

Weinberger, A., Fischer, F., 2006. A framework to analyze argumentative knowledge construction in computer-supported collaborative learning. Computers & education 46, 71–95.

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, 127–147.

White, B., Frederiksen, J., Collins, A., 2009. 10 the interplay of scientific inquiry and metacognition. Handbook of metacognition in education , 175.

White, J., Fu, Q., Hays, S., Sandborn, M., Olea, C., Gilbert, H., Elnashar, A., Spencer-Smith, J., Schmidt, D.C., 2023a. A prompt pattern catalog to enhance prompt engineering with chatgpt. arXiv preprint arXiv:2302.11382 .

White, J., Hays, S., Fu, Q., Spencer-Smith, J., Schmidt, D.C., 2023b. Chatgpt prompt patterns for improving code quality, refactoring, requirements elicitation, and software design. arXiv preprint arXiv:2303.07839 .

Wing, J.M., 2006. Computational thinking. Communications of the ACM 49, 33–35.

Winne, P.H., 2010. Bootstrapping learner's self-regulated learning. Psychological Test and Assessment Modeling 52, 472.

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.

Wise, A.F., Knight, S., Shum, S.B., 2021. Collaborative learning analytics. International handbook of computer-supported collaborative learning , 425–443.

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