A systematic approach for analyzing students' computational modeling processes in C2STEM

Nicole Hutchins[1], Gautam Biswas[1], Shuchi Grover[2], Satabdi Basu[3], and Caitlin Snyder[1]

[1] Vanderbilt University, Nashville TN 37212, USA
[2] Looking Glass Ventures, Palo Alto CA 94306, USA
[3] SRI International, Menlo Park CA 94025, USA
nicole.m.hutchins@vanderbilt.edu

Abstract. Introducing computational modeling into STEM classrooms can provide opportunities for the simultaneous learning of computational thinking (CT) and STEM. This paper describes the C2STEM modeling environment for learning physics, and the processes students can apply to their learning and modeling tasks. We use an unsupervised learning method to characterize student learning behaviors and how these behaviors relate to learning gains in STEM and CT.

Keywords: Learning by modeling, Computational model-building, STEM+CT.

1 Introduction

Modeling is fundamental to science. The Next Generation Science Standards (NGSS) [12] have reinforced the importance of model-based STEM learning to engage students in authentic STEM practices. Our Collaborative Computational STEM (C2STEM) [8] learning environment provides opportunities for students to construct computational models in STEM domains (e.g., [5, 16, 17]) and use these models for problem solving [1, 18]. Such "constructionist" approaches have helped students learn STEM and computational thinking (CT) concepts and practices [16, 3], but some students face difficulties in translating STEM knowledge into computational models [5]. Therefore, students' learning and model building processes merit further investigation.

This paper adopts an exploratory approach to characterize students' learning and model building processes in C2STEM. We apply hierarchical clustering on students' activity data to address the research questions: (1) What patterns of behavior do learners exhibit during computational modeling tasks in a science domain? and (2) What can we glean from these patterns about student learning of science and CT?

2 Background

Our learning-by-modeling paradigm helps students learn by developing, testing, and refining computational models. Such modeling environments provide mechanisms for students to work with multiple representations, receive rapid feedback through the visualization of model behaviors [5, 11], and engage in CT practices [18]. Classroom studies conducted with systems such as CTSiM [3], ViMap [14] and CT-STEM [10] have produced successful summative learning results [5, 15, 18]. We aim to extend this work by analyzing students' model building processes, including impact on learning gains.

Early efforts in the analysis of log data from students' programming process focused on methods to quantify students' modeling progress at each model revision by calculating the distance between the student and expert model [1]; identify program states and assess the likelihood of reaching a "sink" state in which a student was likely to get stuck [5]; and apply exploratory data-driven approaches to design partial solution feedback [13]. In this work, we used unsupervised learning to closely examine the processes students used towards mutually supportive learning of physics and CT, and made attempts to relate their learning performance to groups of student behaviors (e.g., [4, 19]).

Fig. 1. A completed C2STEM model incorporating DSML blocks.

3 The C2STEM Environment

C2STEM scaffolds students' model-building by creating a block-based DSML [8] that provides domain-relevant variables (e.g., acceleration and velocity), and explicit constructs (blocks) for initializing and updating the values of these variables (see Figure 1). This supports exploratory learning by allowing students to execute their developing models and observe the behaviors generated using animations and data tools [8]. While an initialization block (e.g., *green_flag*) is common across block-based environments, we provide additional scaffolding by explicitly providing a *simulation_step* block to help students separate initialization steps from the dynamic update step. In contrast to equation-based modeling, this sets up a temporal *step-by-step* approach to modeling to gain a better understanding of how the behavior of a system evolves over time.

4 Methods

Thirty-five middle school students worked on a 1D motion module in C2STEM that consisted of a training unit and 4 modeling tasks. We used a summative assessment adapted from other studies to measure disciplinary knowledge in physics [2, 7] and CT [1, 6]. Normalized learning gains calculated using $\frac{\text{Posttest} - \text{Pretest}}{\text{Max Possible Score} - \text{Pretest}}$.

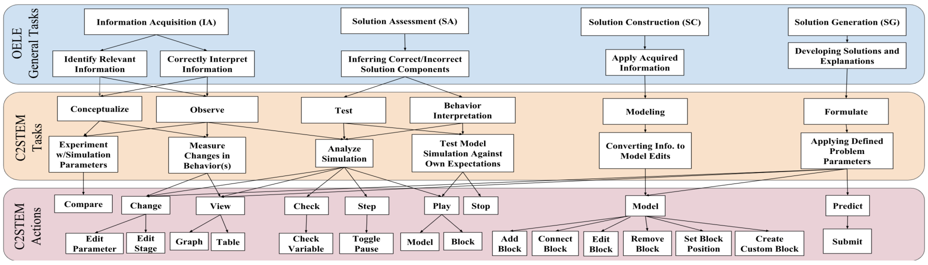

Fig. 2. C2STEM Task Model.

We performed cluster analysis to characterize students' model building behaviors based on actions employed on a constant velocity task (Figure 1). We analyzed data for 29 students, excluding data from students who did not complete either the pre-test or post-test or performed less than five actions. Student actions were recorded in log files with timestamps. We extend a task model developed in our previous work [1] (Figure 2) to interpret students' model building actions. The lowest level captures the discrete model building actions possible, the middle associates a specific purpose for the actions as C2STEM subgoals and the top provides more generic labels to the actions, typically useful for understanding student behaviors across multiple learning environments.

The following features helped cluster students by their model-building activities:

1. *Ratio of total simulations runs to total actions performed (RTP)*: Frequency of SA operations performed.
2. *Ratio of data tool access to total number of simulation runs* (*RDT)*: Frequency of IA and/or SA operations performed.
3. *Average time per access of data tools* (*TDT)*: IA/SA related actions.
4. *Average number of actions between simulation runs (ABP)*: Average size of SC tasks; actions between plays imply a construction process influenced by debugging.
5. *Number of blocks Under Green Flag (NBG)*: a SC task related to variable initialization demonstrating conceptual understanding of problem domain.
6. *Number of blocks in simulation step construct (NST)*: Updating functions (in SC).

We derived a dendrogram structure using the UPGMA hierarchical clustering scheme [9]. The maximum distance between levels heuristic was used to determine the cut-off level and the number of clusters formed. The groups were characterized by distinguishing features, which were then used to explain groups' pre-post learning gains.

5 Results

Table 1. Characterizing clusters based on frequency (*mean*, *sd*) of features.

GR	RTP (SA)	ABP (SA/SC)	RDT (IA/SA)	TDT (IA/SA)	NBG (SC)	NST (SC)
1	0.29 (0.09)	1.96 (0.5)	0.05 (0.05)	4,946.1 (6070.7)	4.25 (0.96)	0.25 (0.5)
2	0.24 (0.03)	2.55 (0.52)	0.24 (0.06)	17,007.4 (20,436.7)	3 (0)	7 (2)
3	0.31 (0.12)	2.14 (0.0)	0.05 (0.05)	21,822.7 (30,106.5)	3.75 (0.34)	8 (0)

Summative assessment results showed that normalized learning gains were *statistically significant*, with t-tests in Physics ($p = 0.009$) and CT ($p = 0.0001$). Cluster analysis produced three distinct groups. Group 1 achieved the highest learning gains in CT [0.50 (0.19)] and moderate Physics learning gains [0.21 (0.23)]. Group 1 is defined by their minimal use of the data tools, highest number of initialization blocks, and very little in terms of update actions to generate dynamic behavior (SC actions). This group also had the least amount of actions between plays (ABP) and the second highest ratio of total plays to total actions (RTP). This may indicate their reliance on trial and error. Given the significant CT learning gains and trial and error approach, we conjecture that these results suggest a focus on programming.

Group 2 achieved the highest learning gains in Physics [0.31 (0.17)] and lowest CT gains [0.22 (0.25)]. Group 2 used data tools (RDT) the most, and had the second largest time usage (TDT). The group had few initialization blocks, forgetting to initialize the simulation step size block [*set delta t to [n] seconds*]. This may have impacted their ability to interpret results from the data tools (for instance, setting delta-t to 1 second would have resulted in variable values updating as integers. Finally, this group had the highest ABP and lower RTP indicating the least amount of testing, implying possible weakness in CT practices such as debugging (as indicated by their low CT gains).

A review of the clustering dendrogram indicates that at the next largest distance, Group 3 breaks into 1 outlier and two subgroups. Subgroup 1 showed higher physics gains [0.23 (2.56)], but lower CT gains of [0.37 (0.36)] (markedly higher than Group 2). All students in this group utilized the data tools, with the highest average TDT and implemented the highest, indicating a more systematic debugging process. Subgroup 2 demonstrated moderate Physics gains, 0.21 (0.16) and higher CT gains, 0.50 (0.16). Their feature values indicate a similar trial and error approaches to Group 1, with low ABP and high RTP, but differences in SC actions may provide useful information into how this approach may impact Physics learning.

6 Discussion and Conclusions

This paper presents initial analyses in linking students' model building behaviors to their pre-post assessment scores. High performers showed better ability to model the update functions. Although exploratory, this work provides unique insights and approaches to the evaluation of block-based computational model building processes in STEM classrooms. As next steps, we are continuing our pattern analysis with larger student populations across different science topics. In addition, we are building more sophisticated logging mechanisms to better understand synergistic learning processes and design adaptive feedback to help students overcome their conceptual difficulties.

Acknowledgments. We thank Marian Rushdy, Naveed Mohammed, and our other collaborators at Vanderbilt University, Stanford University, Salem State University, SRI International, and ETS. This research is supported by NSF grant #1640199.

References

1. 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.
2. Basu, S., McElhaney, K., Grover, S., Harris, C., and Biswas, G. (2018). A principled approach to designing assessments that integrate science and computational thinking. Proceedings of ICLS'18.
3. Basu, S., Dickes, A., Kinnebrew, J.S., Sengupta, P., & Biswas, G.: CTSiM: A Computational Thinking Environment for Learning Science through Simulation and Modeling. Conference on Computer Supported Education, pp. 369-378, Germany (2013)
4. Berland, M., Martin, T., Benton, T., Smith, C.P., and Davis, D.: Using learning analytics to understand the learning pathways of novice programmers. Journal of the Learning Sciences 22(4), 564-599 (2013)
5. Blikstein, P., Worsley, M., Piech, C., Sahami, M., Cooper, S., and Koller, D.: Programming pluralism: Using learning analytics to detect patterns in the learning of computer programming. Journal of the Learning Sciences, 23(4), 561–599 (2014)
6. Grover, S., Jackiw, N., & Lundh, P.: Concepts before coding: non-programming interactives to advance learning of introductory programming concepts in middle school. Computer Science Education, (2019). DOI: 10.1080/08993408.2019.1568955
7. Hestenes, D., Wells, M., Swackhamer, G.: Force concept inventory. The physics teacher, 30(3), 141-158 (1992)
8. Hutchins, N., Biswas, G., Maroti, M., Broll, B., and Ledezci, A. (2018). C2STEM: A design-based approach to a classroom-centered OELE. Proceedings of AIED '18.
9. Johnson, S. C.: Hierarchical clustering schemes. Psychometrika, 32(3), 241-254 (1967)
10. Jona, K., Wilensky, U., Trouille, L., Horn, M. S., Orton, K., Weintrop, D., Beheshti, E.: Embedding computational thinking in science, technology, engineering, and math (CT-STEM). In Future Directions in Computer Science Education Summit Meeting, Orlando, FL (2014)
11. Jonassen, D., Strobel, J., Gottdenker, J.: Model building for conceptual change. Interactive Learning Environments, 13(1-2), 15-37 (2005)
12. NGSS Lead States: Next Generation Science Standards: For states, by states. National Academies Press, Washington, DC (2013)
13. Piech, C., Huang, J., Nguyen, A., Phulsuksombati, M., Sahami, M., and Guibas, L.: Learning program embeddings to propagate feedback on student code. In Proceedings of the 32nd International Conference on Machine Learning. pp. 1093–1102, Lille, France (2015)
14. Sengupta, P., Dickes, A., Farris, A. V., Karan, A., Martin, D., & Wright, M.: Programming in K-12 science classrooms. Communications of the ACM, 58(11), 33-35 (2015)
15. Sengupta, P., Farris, A. V., Wright, M.: From agents to continuous change via aesthetics: learning mechanics with visual agent-based computational modeling. Technology, Knowledge and Learning, 17(1-2), 23-42 (2012)
16. Sengupta, P., Kinnebrew, J.S., Basu, S., Biswas, G., Clark, D.: Integrating Computational Thinking with K-12 Science Education Using Agent-based Computation: A Theoretical Framework. Education and Information Technologies, 18(2), 351-380 (2013)
17. Shen, J., Lei, J., Chang, H. Y., Namdar, B.: Technology-enhanced, modeling-based instruction (TMBI) in science education. In: J. M. Spector, M. D. Merrill, J. Elen, & M. J. Bishop (Eds.), Handbook of research on educational communications and technology, pp. 529-540. Springer, New York, NY (2014)

18. Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., Wilensky, U.: Defining Computational Thinking for Mathematics and Science Classrooms. Journal of Science Education and Technology, 25(1), 127-147 (2016)
19. Werner, L., McDowell, C., and Denner, J.: A first step in learning analytics: Pre-processing low-level Alice logging data of middle school students. Journal of Educational Data Mining 5(2) 11–37 (2013)