A Framework For Hypothesis-Driven Approaches To Support Data-Driven Learning Analytics In Measuring Computational Thinking In Block-Based Programming

Shuchi Grover
SRI International, Menlo Park, CA
shuchi.grover@sri.com

Marie Bienkowski
SRI International, Menlo Park, CA
marie.bienkowski@sri.com

Satabdi Basu
SRI International, Menlo Park, CA
satabdi.basu@sri.com

Michael Eagle
HCII, Carnegie Mellon University
meagle@cs.cmu.edu

Nicholas Diana
HCII, Carnegie Mellon University
ndiana@cs.cmu.edu

John Stamper
HCII, Carnegie Mellon University
jstamper@cs.cmu.edu

ABSTRACT

K-12 classrooms use block-based programming environments (BBPEs) for teaching computer science and computational thinking (CT). To support assessment of student learning in BBPEs, we propose a learning analytics framework that combines hypothesis- and data-driven approaches to discern students' programming strategies from BBPE log data. We use a principled approach to design assessment tasks to elicit evidence of specific CT skills. Piloting these tasks in high school classrooms enabled us to analyze student programs and video recordings of students as they built their programs. We discuss *a priori* patterns derived from this analysis to support data-driven analysis of log data in order to better assess understanding and use of CT in BBPEs.

1. INTRODUCTION

Most K-12 computer science (CS) courses teach programming to support learning of computational thinking (CT) practices such as decomposing problems, debugging, and use of CT concepts to create computational solutions. However, programming is, and has been, difficult for novices to learn [5], and assessing K-12 students' learning is typically done manually by checking students' final programs, giving an incomplete picture of students' CT skills. Examining **process** gives a more complete picture [2].

We present a theoretical framework that researchers can use to design measurement systems for block-based programming environments (BBPEs) for research or application. Using this framework, we analyzed log data from a previously designed assessment of middle school CT in the Alice BBPE; designed new tasks based on the Evidence Centered Design (ECD) framework (a principled approach to guide assessment design [3]), and derived *a priori* hypothesis-driven patterns. We describe how the framework can inform future efforts that blend hypothesis- and data-driven approaches for measuring CT skills.

2. RELATED WORK

Programming is a complex activity that involves understanding a problem as a computational task, mapping a design for the program, drawing on problems previously programmed that have a similar structure, instantiating abstract program patterns, coding the program, and then testing and debugging. Past research relies on examining the finished programs for use of programming constructs, however, learning analytics (LA) approaches [1] offer better ways to analyze student understanding, misconceptions, and steps to a solution using data from digital environments such as number of actions in students' programs and number of successful and unsuccessful program compilations [2]. Clustering techniques [3] led to various programmer behavior profiles, and unsupervised methods were used to derive program-state patterns and state transitions to predict success outcomes [4]

Early efforts have mostly analyzed log data post-hoc, looking for static constructs or patterns largely from the "bottom up" [8] using data-driven LA. New *blended LA* assess students' learning processes in various digital learning environments by combining ECD and LA for hypothesis-driven generation of *a priori* patterns about learner actions [6]. ECD focuses on three related models: student (what are targeted cognitive constructs?), task (what activities allow students to demonstrate cognitive constructs?), and evidence (what data provide evidence of cognitive constructs?). ECD helps connect important constructs that we want to measure with observable behaviors (including patterns of learner actions). Also, importantly, evidence is obtained by deliberately putting students in situations or tasks that will elicit the needed evidence. Once semantically meaningful patterns are defined *a priori,* data mining techniques can be used to analyze the patterns further.

3. METHODOLOGY

This work is part of a broader effort to study learner behavior in BBPEs, specifically, ***What patterns of behavior in data logs from BBPEs provide evidence of learners' use of CT concepts and practices?*** In Phase 1, we analyzed a dataset from an assessment task designed and used in prior research [7]. 118 females and 202 males aged 10 to 14 years completed the 30-minute task which involved modifying existing code. Students' programs and Alice log data were collected, and the programs were scored manually using a rubric for algorithmic thinking and abstraction. We had complete data for 229 students. We applied ECD to "reverse engineer" this task into specific CT concepts and skills and give evidence of what those might look like in log files. We also compared action sequences between students who scored high and low (relative to the median) to determine commonality of sequences for each group. We found sequences that were significantly more common among students with high grades and one sequence that occurred significantly more frequently for students with low grades. Further analysis showed that a higher

LAK '17, March 13-17, 2017, Vancouver, BC, Canada
ACM 978-1-4503-4870-6/17/03.
http://dx.doi.org/10.1145/3027385.3029440

number of code edit actions as well as more frequent testing were associated with higher final scores. Also, students who made more changes to their code tended to test their program more frequently *however, whether students tested their programs frequently, or after working on a considerable part of the program, did not seem to have a bearing on their final scores*. In this way, we gained insights into interpreting student actions from logs. In Phase 2, we applied ECD for forward design of a more complex task to generate richer ***process*** data to observe repeated use of constructs and CT practices. We designed two new Alice assessment tasks aligned with focal CT concepts and practices. These tasks were piloted in two high school introductory CS classrooms with 27 and 28 students. Data included final Alice files and log data for all students and screen recordings for 6 students. Analysis of logs revealed similar issues that students struggled with in both tasks: hard-wired vs. general solutions, improper termination conditions, parallel vs. sequential execution, effective solution decomposition, and appropriate random number use.

Analyses of screen captures from the 6 students using a ‘process over product’ lens to assess CT practices showed that some students created their own methods, tested the methods in isolation, and then used the methods in the main method. These practices demonstrate abstraction, modularization, and testing in parts, and could serve as useful patterns to search for in students’ log data as evidence for CT skills. In addition, we noticed certain phases during students’ programming process when a student was unable to progress. The student added, deleted, or reorganized existing actions and repeatedly tested the program after each small edit. While repeatedly testing and editing is not a practice we need to discourage, it does become problematic when a student cannot progress towards the task goal even after repeated editing and testing. Such circumstances can easily lead to frustration and loss of engagement, and can thus serve as good candidates for potential patterns to be detected as students work on their assessment tasks.

3.1 Hypothesis-Driven Framework

The research described applies ECD to conduct thoughtful and deliberate hypothesis-driven analyses. These analyses reveal patterns of behavior that complement data-driven findings of student programming actions. Our work helps us articulate a preliminary framework for effectively using a hybrid approach to interpret student actions in log data from BBPEs in general. The through line from the ECD domain modeling stage to the patterns derived in Section 3.2 helps us connect log data entries to CT skills and practices that we aim to measure (both formatively and summatively) in order to make claims about student understanding of CT skills. The framework (Fig. 1) describes an iterative process that begins with articulating important CT concepts and practices. Careful design of tasks put students in situations that evoke behaviors to provide potential observables of these concepts and practices. Detailed code analysis of varied solutions reveals students’ use of constructs (correct or otherwise) and approaches to solutions. Similarly, analyzing data from observations reveals aspects of students’ actions that are never seen in the final program. These can reveal student misunderstanding of concepts even if the final solution seemingly demonstrates correct usage. Combined qualitative analyses of the solutions and process of a designed task provide a deeper understanding than is possible from data-driven analytics alone, including potential code sequences that map to practices that could be detected in logs. These hypotheses lay the foundation for detectors for these patterns and provide a richer interpretation of student process in BBPEs.

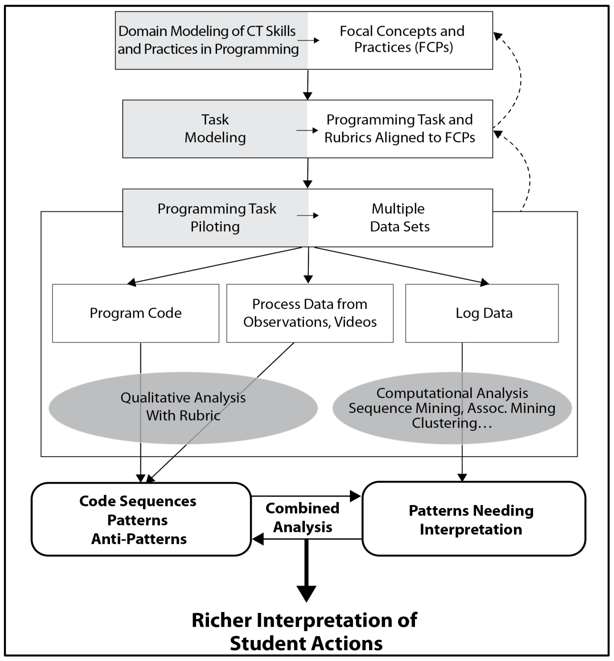
Domain Modeling of CT Skills and Practices in Programming
Focal Concepts and Practices (FCPs)
Task Modeling
Programming Task and Rubrics Aligned to FCPs
Programming Task Piloting
Multiple Data Sets
Program Code
Process Data from Observations, Videos
Log Data
Qualitative Analysis With Rubric
Computational Analysis Sequence Mining, Assoc. Mining Clustering…
Code Sequences Patterns Anti-Patterns
Combined Analysis
Patterns Needing Interpretation
Richer Interpretation of Student Actions

Figure. 1: Proposed Framework for Hypothesis-driven Analyses to Support Data-Driven Analytics

4. CONCLUSION & NEXT STEPS

Our emergent framework for using hypothesis-driven analyses to support data-driven learning analytics leads to better interpretation of student actions to assess skills in BBPEs. Currently, in Phase 3, we are using and refining our framework. We are gathering program files and log data from ~100 students in three high school classrooms in Fall/Winter of 2016/17, and classroom observations and screen recordings with interviews from a few students.

5. ACKNOWLEDGMENTS

We gratefully acknowledge grant support from NSF IIS-1522990, and Sangeeta Bhatnagar for her support in conducting classroom research.

6. REFERENCES

[1] Baker, R., & Siemens, G. 2014. Educational data mining and learning analytics. In K. Sawyer (Ed.), *Cambridge Handbook of the Learning Sciences*.

[2] Berland, M., Martin, T., Benton, T., Petrick Smith, C., & Davis, D. 2013. Using Learning Analytics to Understand the Learning Pathways of Novice Programmers. *JLS, 22*(4), 564–599.

[3] Mislevy, R.J. & Haertel.G. 2006. Implications of Evidence-Centered Design for Educational Testing. *Educational Measurement: Issues and Practice*, *25*(4), 6-20.

[4] Piech, C., et al. 2012. Modeling how students learn to program. In *Proceedings of the 43rd SIGCSE* (pp. 153-160). ACM.

[5] Robins, A., Rountree, J., & Rountree, N. 2003. Learning and teaching programming: A review and discussion. *Computer Science Education*, 13(2), 137-172.

[6] Rupp, A. A., et al. 2012. Putting ECD into Practice: The Interplay of Theory and Data in Evidence Models within a Digital Learning Environment. *JEDM, 4*(1), 49–110.

[7] Werner, L., Denner, J., Campe, S., & Kawamoto, D.C. 2012. *The Fairy Perf. Assessment: Measuring computational thinking in middle school*. In *Proceedings of the 43rd SIGCSE*. ACM.

[8] Winne, P. H., & Baker, R. S. 2013. The potentials of educational data mining for researching metacognition, motivation and self-regulated learning. *Journal of Educational Data Mining, 5*(1).