Analyzing Students' Computational Models as they Learn in STEM Disciplines

Mr. Anton Dukeman, Vanderbilt University
Mr. Shashank Shekhar, Vanderbilt University
Mr. Faruk Caglar, Vanderbilt University
Dr. Aniruddha Gokhale, Vanderbilt University

Aniruddha Gokhale is an Associate Professor of Computer Science and Engineering in the Dept of Electrical Engineering and Computer Science at Vanderbilt University, Nashville, TN, USA. Prof. Gokhale got his BE (Computer Engineering) from Pune University, Pune, India in 1989; MS (Computer Science) from Arizona State University, Tempe, AZ in 1992; and PhD (Computer Science) from Washington University in St. Louis, St. Louis, MO in 1998. Prior to his current position at Vanderbilt University, he was a Member of Technical Staff at Lucent Bell Labs. He is a Senior Member of both the IEEE and ACM.His research interests are in solving distributed systems challenges for real-time and embedded systems through effective software engineering principles and algorithm development. He is applying these expertise to develop an effective, cloud-based and ubiquitous infrastructure for scalable, collaborative STEM education.

Prof. Gautam Biswas, Vanderbilt University

Gautam Biswas is a Professor of Computer Science, Computer Engineering, and Engineering Management in the EECS Department and a Senior Research Scientist at the Institute for Software Integrated Systems (ISIS) at Vanderbilt University. He has an undergraduate degree in Electrical Engineering from the Indian Institute of Technology (IIT) in Mumbai, India, and M.S. and Ph.D. degrees in Computer Science from Michigan State University in E. Lansing, MI.

Prof. Biswas conducts research in Intelligent Systems with primary interests in hybrid modeling, simulation, and analysis of complex embedded systems, and their applications to diagnosis, prognosis, and fault-adaptive control. As part of this work, he has worked on fault diagnosis and fault-adaptive control of secondary sodium cooling systems for nuclear reactors, automobile engine coolant systems, fuel transfer systems for aircraft, Advanced Life Support systems and power distribution systems for NASA. He has also initiated new projects in health management of complex systems, which includes online algorithms for distributed monitoring, diagnosis, and prognosis. More recently, he is working on data mining for diagnosis, and developing methods that combine model-based and data-driven approaches for diagnostic and prognostic reasoning. This work, in conjunction with Honeywell Technical Center and NASA Ames, includes developing sophisticated data mining algorithms for extracting causal relations amongst variables and parameters in a system. For this work, he recently received the NASA 2011 Aeronautics Research Mission Directorate Technology and Innovation Group Award for Vehicle Level Reasoning System and Data Mining methods to improve aircraft diagnostic and prognostic systems.

In other research projects, he is involved in developing simulation-based environments for learning and instruction. The most notable project in this area is the Teachable Agents project, where students learn science by building causal models of natural processes. More recently, he has exploited the synergy between computational thinking ideas and STEM learning to develop systems that help students learn science and math concepts by building simulation models. He has also developed innovative educational data mining techniques for studying students' learning behaviors and linking them to metacognitive strategies. His research has been supported by funding from NASA, NSF, DARPA, and the US Department of Education. His industrial collaborators include Airbus, Honeywell Technical Center, and Boeing Research and Development. He has published extensively, and has over 300 refereed publications.

Dr. Biswas is an associate editor of the IEEE Transactions on Systems, Man, and Cybernetics, Prognostics and Health Management, and Educational Technology and Society journal. He has served on the Program Committee of a number of conferences, and most recently was Program co-chair for the 18th International Workshop on Principles of Diagnosis and Program co-chair for the 15th International Conference on

Artificial Intelligence in Education. He is currently serving on the Executive committee of the Asia Pacific Society for Computers in Education and is the IEEE Computer Society representative to the Transactions on Learning Technologies steering committee. He is also serving as the Secretary/Treasurer for ACM Sigart. He is a senior member of the IEEE Computer Society, ACM, AAAI, and the Sigma Xi Research Society.

Dr. John S Kinnebrew, Vanderbilt University

Analyzing Students' Computational Models as they Learn in STEM Disciplines (Work in Progress)

Introduction

The 21st century workplace places a heavy emphasis on competence in STEM disciplines, but unfortunately the US is lagging behind a number of the advanced countries in STEM competency at all levels.[1] Therefore, more effective methods need to be developed for students to gain a deeper understanding and develop better problem solving skills through middle school and high school education. Among the ways STEM education can be made more interesting and relevant is by tying it to real-world problems and incorporating active design and engineering principles in to the learning process. We have created the Challenge-based Collaborative Community-centered STEM (C^3STEM) learning environment, where students collaborate to solve challenging real-world problems. The project incorporates modeling and simulation activities with the Computational Thinking through Simulation and Modeling (CTSiM) environment to help students learn fundamental STEM concepts.[2–4] Students build visual models of vehicles and traffic operations then simulate their models through model transformations and execution with the NetLogo[5] simulation engine. In this paper, we present measures and a preliminary analysis to assess the evolution of students' CTSiM programs/models in the C^3STEM project. In C^3STEM, students progress from individual vehicle models to building more advanced traffic flow models through city streets and intersections. By building and refining these models, students learn science curriculum fundamentals and mathematical modeling principles, while applying them to real-world contexts.

Some of the more well-known comparison metrics, such as the bag of words score and abstract syntax tree edit distance, allow us to assign a similarity score between each version of a student's model and the expert model, which is not made available to the students. However, students can compare their model behaviors with the expert model behaviors. Tracking model changes over time allows us to better understand the evolution of students' approach to the modeling and simulation tasks, as well as their understanding of physics, mathematics, and computational thinking constructs. Further, detecting errors in student models facilitates automated scaffolding of the student's learning of computational thinking skills and domain specific knowledge.

Background

C^3STEM incorporates two core learning environments based on modeling and simulation: CTSiM[2,3] (Computational Thinking in Simulation and Model-Building) for micro-level modeling of vehicle and traffic operation, and C^2SuMo (Cloud-based, Collaborative, Scaled-up Modelling environment) for macro-level modeling using a Google Maps interface along with the high-fidelity Simulation of Urban MObility (SUMO)[6] traffic simulator for macro-level simulation. Students start with CTSiM to learn how traffic operates at the micro level, focusing on concepts including acceleration/deceleration, individual car behavior at a stop sign, and behavior of multiple cars at an intersection with stoplights. CTSiM provides a visual modeling and simulation environment

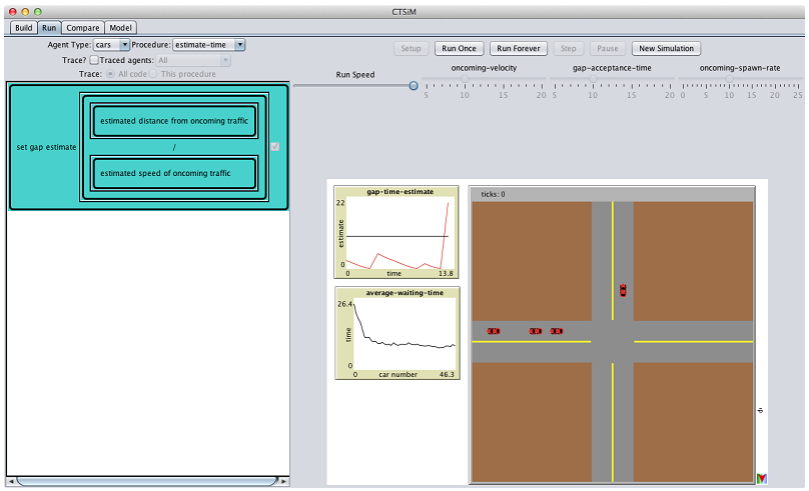

Figure 1: CTSiM with Driver Behavior Domain

that has been employed to design a progression of C^{3}STEM micro-level modeling units: basic mechanics (focusing on the relationship between position, velocity, and acceleration), driver behavior (focusing on how driver's can estimate the extent of gaps in oncoming traffic to perform unprotected left turns, see Figure 1), and intersection behavior (focusing on traffic flow and the effects of stoplights at an intersection). After these CTSiM modeling and simulation activities, students work collaboratively to solve a challenge problem to improve traffic flow with SUMO, which abstracts away the basic kinematics modeling but can simulate large traffic flows on real (GIS) maps of local roads.

For students to succeed in these complex modeling and simulation activities requires significant scaffolding, currently provided in the C^{3}STEM project by the researchers. However, to scale up the usage of C^{3}STEM in full classrooms requires automating some of the scaffolding within the learning environment itself. In this paper, we present the first step towards this dynamic scaffolding by demonstrating automated methods to measure and analyze the student's progress in model design and revision. This work is based on extending techniques for analyzing code in more traditional programming environments. In particular, researchers have used a bag-of-words-based metric and analysis of abstract syntax tree (AST) changes to assess the similarity of code segments as students learn to program.[7] We use techniques derived from both of these methods to analyze behavior in our visual programming environment.

Modeling

Being able to analyze a student's progression of models as they work on a unit is an important first step in understanding their difficulties and then scaffolding their learning. The Bag of Words (BoW) approach determines if the student's model has the same building blocks as the expert model, while the Tree Edit Distance (TED) measure can describe the extent to which the blocks are correctly aligned to produce a model corresponding to the expert model.

We define the bag of words for a model as the set of block names that exist in the abstract syntax

tree of the student's model. Using Figure 2 as an example, the bag consists of {set, $position$, plus, $position$, times, $velocity$, Δt }. From this we have defined a model accuracy measure based on the BoW approach. First, we calculate a percentage correct score, which describes how many of the blocks in the expert model are also in the student's model as shown in equation 1. Next, we calculate a normalized incorrectness score, which describes how many extraneous blocks the model has with respect to the number of blocks required in the correct model, which is shown in equation 2. Finally, the BoW measure combines these two scores as a distance metric. To calculate this BoW distance measure, the correctness and incorrectness scores are used as dimensions in a two-dimensional vector (i.e. $< percentCorrect, normalizedIncorrect >$). By this definition the expert model is located at $< 1, 0 >$ and the BoW distance measure is then the Euclidean distance between the student's model and the expert model.

$$percentCorrect = \frac{|student \cap expert|}{|expert|} \tag{1}$$

$$normalizedIncorrect = \frac{|student| - |student \cap expert|}{|expert|} \tag{2}$$

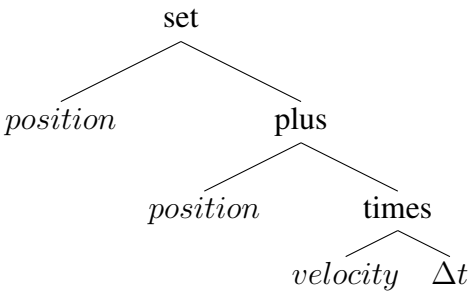

Figure 2: Example expert AST

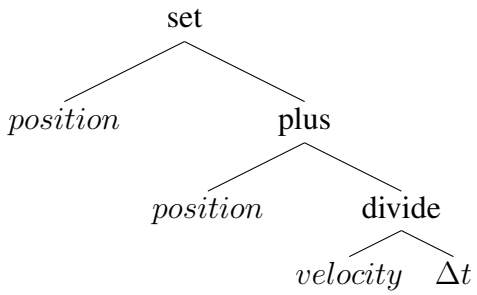

Figure 3: Example Student AST

The second technique for analyzing students' models is based on the edit distance between abstract syntax tree representations. This measure defines the distance between two models as the number of operations (node renaming, insertion, and deletion) that must be performed on the abstract syntax tree in order to convert one model to another. This measure places an emphasis on how blocks are combined to produce the model. Figure 4 illustrates the results of applying each of these measures for comparing the example models in Figure 2 and 3.

In general, there are several ways to construct correct (and behaviorally equivalent) models. Therefore, additional transformations could be applied to achieve a canonical form of the models or the

Bag of Words Percent Correct	$\frac{6}{7} = 0.857$
Bag of Words Normalized Incorrect	$\frac{1}{7} = 0.143$
Bag of Words distance	$\sqrt{(1 - 0.857)^2 + (0 - 0.143)^2} = 0.202$
Tree Edit Distance	1.000

Figure 4: Various metrics for comparing Figure 2 and 3

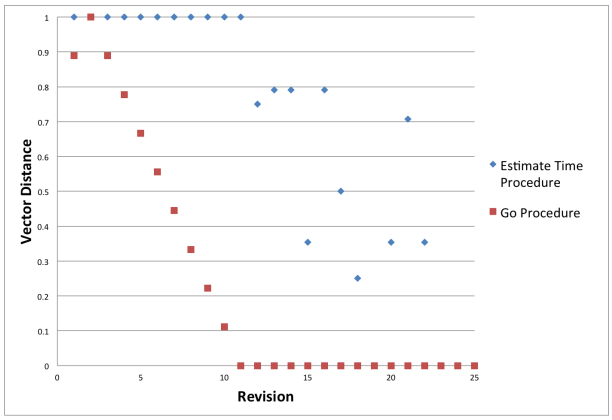

Figure 5: Bag of Words Distance for Student B

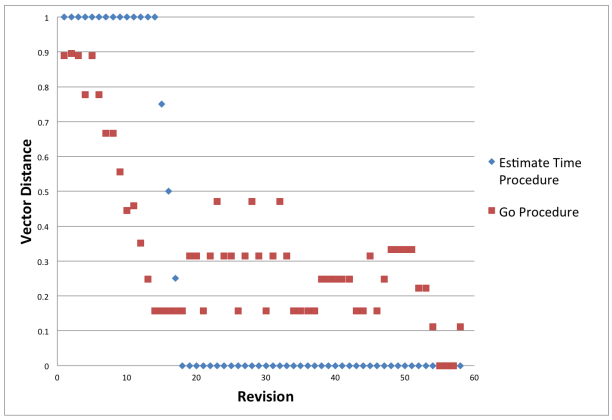

Figure 6: Bag of Words Distance for Student E

modeling language and task can be designed such that there are only a small number ways to construct the correct behavior. For example, $a > b$ and $b < a$ will operate exactly the same, but their AST representations are different. In C^{3}STEM units, the modeling language for each is sufficiently targeted to the domain that we can easily construct a canonical form of the models in the AST representation.

Analysis and Discussion

Data was collected from seven high school students working on a series of C^{3}STEM units during an internship at Vanderbilt University. In this section, we analyze data from two students working on the second unit to illustrate the complementary aspects of the two measures defined in the previous section. In this unit, students modeled a car making an unprotected left turn, including estimating the time available for turning in a gap between oncoming cars and comparing this to the time required to make the turn.

The results for Student B's model in this unit are presented in Figures 5 and 7. As shown by the BoW measure in Figure 5, the student had all of the correct building blocks for the "Go" procedure by their 11th revision of the model. However, the TED measure (Figure 7) shows that their model was still not correctly completed. This discrepancy indicates that while the student had the correct blocks in their model a number of them were not correctly ordered or nested. In this case, the problem was an incorrectly nested conditional block, which student B corrected in his penultimate model revision by moving the entire nested conditional block out one level, exactly matching the expert model. Another example illustrating the complementary use of these two measures is Student E's work on this unit. This student got close to the correct set of blocks, as indicated by the BoW measure (Figure 6), about a third of the way through his revisions and fluctuated close to this level for most of the rest of his work on the unit. However, the TED measure (Figure 8) illustrates that the student was primarily rearranging the existing blocks and, in fact, generally causing his model to diverge further from the expert model (indicated by the frequent and mostly increasing changes in TED but less frequent and smaller changes in BoW).

In general, there are several possibilities for what happens to the TED and BoW measures when

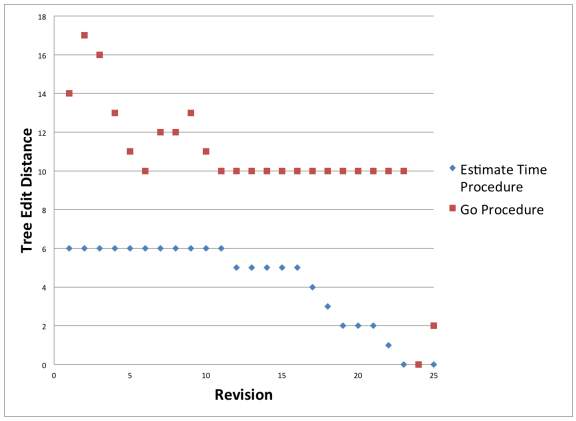

Figure 7: Tree Edit Distance for Student B

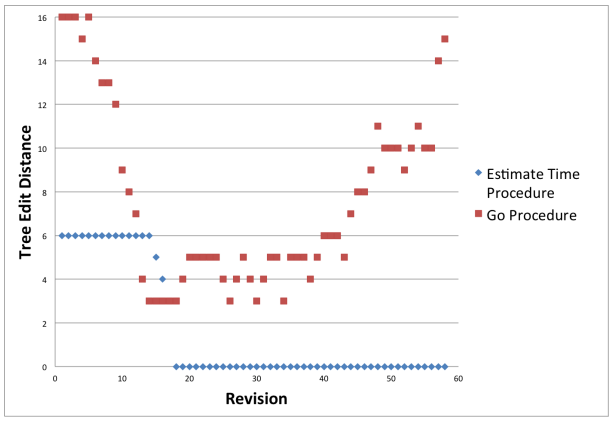

Figure 8: Tree Edit Distance for Student E

a change is made to the model. One possibility is that TED and BoW both decrease (i.e., both indicate that the model has gotten closer to the expert model), which means that an incorrect block was removed or a correct block was added to its correct location (while the reverse is indicated if both measures increase). Another option is that TED increases and BoW decreases, which points to a block that belongs in the model being inserted in the wrong location. When BoW remains constant but TED increases or decreases, it indicates that a block was moved either from a correct to incorrect location or an incorrect to correct location, respectively. With these relationships, we can classify edits and scaffold student activity based on their edits and patterns in their edits.

Future Work

Future work will include integrating dynamic scaffolding informed by these measures into the modeling environment. For example, if the student is going further and further away from the expert model, the system can guide students in the right direction, and provide advice on fixing incorrect portions of the model. Sometimes, the feedback module can play a more active role for students who are struggling by pointing to blocks in the model code that are unnecessary for the current procedure. Future data collection will also provide evidence for the range of editing patterns that occur during student modeling in C^3STEM and their relationship to the metrics discussed in this paper.

Acknowlegements

This work was supported by NSF EAGER grant #1257955.

References

[1] Dana Kelly et al. Performance of U.S. 15-Year-Old Students in Mathematics, Science, and Reading Literacy in an International Context: First Look at PISA 2012. *NCES 2014-024, U.S. Department of Education*, 2013.

[2] Satabdi Basu, Amanda Dickes, John S Kinnebrew, Pratim Sengupta, and Gautam Biswas. Ctsim: A computational thinking environment for learning science through simulation and modeling. In *The 5th International Conference on Computer Supported Education*, 2013.

[3] Pratim Sengupta, John S Kinnebrew, Satabdi Basu, Gautam Biswas, and Douglas Clark. Integrating computational thinking with k-12 science education using agent-based computation: A theoretical framework. *Education and Information Technologies*, 18(2):351–380, 2013.

[4] Anton Dukeman, Faruk Caglar, Shashank Shekhar, John Kinnebrew, Gautam Biswas, Doug Fisher, and Aniruddha Gokhale. Teaching computational thinking skills in c^3stem with traffic simulation. *Human-Computer Interaction and Knowledge Discovery in Complex, Unstructured, Big Data: Vol. 7947. Lecture Notes in Computer Science*, 2013.

[5] Uri Wilensky. NetLogo. *Center for connected learning and computer-based modeling, Northwestern University*, 1999.

[6] Daniel Krajzewicz, Georg Hertkorn, C Rössel, and P Wagner. Sumo (simulation of urban mobility). In *Proc. of the 4th Middle East Symposium on Simulation and Modelling*, pages 183–187, 2002.

[7] Chris Piech, Mehran Sahami, Daphne Koller, Steve Cooper, and Paulo Blikstein. Modeling how students learn to program. In *Proceedings of the 43rd ACM technical symposium on Computer Science Education*, pages 153–160. ACM, 2012.