So, H. J. et al. (Eds.) (2020). Proceedings of the 28th International Conference on Computers in Education. Australia: Asia-Pacific Society for Computers in Education

Using Log and Discourse Analysis to Improve Understanding of Collaborative Programming

Bernard YETT[a*], Caitlin SNYDER[a], Ningyu ZHANG[a], Nicole HUTCHINS[a], Shitanshu MISHRA[a] & Gautam BISWAS[a]
[a]*Vanderbilt University, USA*
*bernard.h.yett@vanderbilt.edu

Abstract: The importance and ubiquity of computing and computational thinking (CT) is leading to re-design of K-12 curricula and the development of appropriate platforms to support computer science instruction. One such environment is the block-based, synchronous programming environment, NetsBlox, that allows for real-time collaboration amongst students. This work presents a novel combination of discourse and activity log analyses to study the collaborative behaviors of K-12 students as they worked on a week-long cybersecurity curriculum. Groups of students were assessed based on pre-post-test learning gains in cybersecurity and CT. We analyze the differences between the collaborative behaviors and discourse of high and low performing groups using case study and differential sequence mining analyses to characterize productive and unproductive collaborative problem solving in programming tasks.

Keywords: computer science education, data-driven analysis, collaboration, robotics

1. Introduction

Collaboration is considered a key computational practice (College Board, 2020) and offers several benefits worthy of further exploration. Previous work has shown that collaboration provides social support, thus allowing a group to learn from each other and overcome difficulties (Saenz-Otero et al., 2010). Werner et al. (2012) found a significant positive relationship between time spent programming with a partner and assessment scores. Despite these, there is evidence of problems that can arise while students work together on programming projects. When pairing students with different experience levels, the more knowledgeable student often assumes full control and leaves their partner confused and disengaged (Braught, MacCormick, & Wahls, 2010). Working with others can be a frustrating, challenging experience that may ultimately undermine motivation (Rogat, Linnenbrink-García, & DiDonato, 2013).

Taken together, it becomes important to design environments that guide students towards the benefits of collaboration while avoiding the pitfalls. To accomplish this, we need an improved understanding of the problem-solving processes groups employ, specifically the interweaving of dialog and actions as they work on their programming tasks. This may reveal how students develop an understanding of the desired computer science concepts and practices, while also exposing their difficulties and how they may overcome them via collaboration. Each mode of interaction requires unique analysis techniques in order to uncover the collaborative processes at play and ensure the success of each group. Our specific approach to collaborative programming analysis integrates discourse, learning, and log data, a combination that is still under-researched.

Cybersecurity was the primary focus of this intervention. It has become an important component of computer science (CS) curricula, but presents complex concepts and can be difficult to grasp when not situated within meaningful contexts (Thompson et al., 2018). Robotics platforms have been successful in delivering cybersecurity curricula to K-12 students (Karaman et al., 2017; Lédeczi et al., 2019; Yett et al., 2020a). These platforms allow students to collaborate to solve larger, more complex problems through co-construction of knowledge (Fischer, Bruhn, Gräsel, & Mandl, 2002; Hmelo-Silver, 2003).

In this study, students used the NetsBlox (Broll et al., 2017) environment to develop secure robot programs. NetsBlox features a collaborative editor where students can code together in a shared and synchronized workspace while working across multiple computers. Students were co-located and used their own laptops as they developed their robot programs, dynamically choosing their own roles in accord with findings related to potential issues of forced turn-taking (Tsan, Lynch, & Boyer, 2018).

In order to investigate the collaborative potential of NetsBlox as a flexible, synchronous, co-located platform, we target the following research questions: *(1) What programming behaviors as observed via differential sequence mining distinguish groups of students that are successful in the assigned tasks versus those that are not? (2) How did the relative presence or absence of collaboration during a specific project affect student understanding as indicated by learning gains?*

RQ1 provides insights into the collaborative programming process students used during the intervention. We hypothesize that groups with greater shared understanding of the task implemented patterns demonstrating systematic development of their code. RQ2 studies the impact of collaboration on learning performance on specific tasks. We hypothesize that groups with greater interactive discourse demonstrated improved collaborative processes targeting a shared understanding.

2. Background

Collaboration is "*a coordinated, synchronous activity that is a result of a continuous attempt to construct and maintain a shared conception of a problem*" (Roschelle & Teasley, 1995, p.70). Successful collaboration requires the co-construction of knowledge (Fischer et al., 2002; Hmelo-Silver, 2003) between group members along with interactions on progress monitoring, providing constructive feedback, and encouraging the contribution of ideas (Garrison & Akyol, 2013; Grau & Whitebread, 2012). To increase understanding of collaborative problem-solving (CPS) processes, researchers have developed several methodologies, such as after-the-fact reviews (Zimmerman & Pons, 1986) and concurrent think-alouds (Branch, 2013) to gain a deeper understanding of problem-solving processes.

However, these methodologies have limitations, such as determining if the student is actually reporting what they thought at the moment and not inferring after the fact, unintentionally biasing the students with leading questions and interrupting the student's thought processes (Branch, 2013; Zimmerman & Pons, 1986). To overcome such limitations, researchers have utilized students' naturally occurring collaborative dialogue to provide insight into their collaborative problem-solving processes. Tsan et al. (2018) performed a qualitative analysis of pairs of students working collaboratively, including observations of how students claimed roles and how equally distributed dialogue was among group members. Zakaria et al. (2019) compared collaborating students' discourse when they either shared a computer or worked side-by-side on separate computers on a shared project, finding no significant differences in the quantity of collaborative talk between the two configurations.

While this dialogue is a rich resource, analysis can be time-intensive and may miss out on relevant information. For example, students may apply signifiers such as "this" or "that" in their discourse, referring to specific program components that are not immediately clear when looking at discourse transcriptions alone. Including students' action logs can overcome such limitations. Kinnebrew, Segedy, and Biswas (2017) contributed a framework combining analytic measures (model-driven metrics) with patterns derived from students' action sequences (pattern mining) during model building tasks to understand students' problem-solving processes and the difficulties they face when working in open-ended learning environments (OELEs). Werner, McDowell, and Denner (2013) provided a tool to combine low-level logs into interpretable problem-solving strategies.

However, students typically share one computer during co-located collaborative programming, and this makes it difficult to distinguish one student's actions from another. A multimodal analysis approach that includes their discourse as well as individual actions in the system allows for a deeper analysis of collaborative problem solving. SLAM-KIT (Noroozi et al., 2019) is recent work aimed at aiding researchers as they combine multiple data streams and seek to better understand collaborative learning. Grover et al. (2016b) present a pilot study for multimodal analysis within a collaborative setting. Vrzakova et al. (2020) considered screen activity, discourse, and body motion as unimodal primitives and extended them to their multimodal combinations.

Multimodal analysis creates an opportunity to further understand collaborative programming. Research on students' collaborative programming has mostly focused on pair programming. However, some take-aways can apply to other forms of collaborative programming. Shi, Shah, Hedman, and O'Rourke (2019) describe effective collaboration, in the context of programming, occurring when all group members are actively involved in all aspects of the problem process, particularly the construction of the solution. During successful collaboration, group members share the responsibilities of discussion, planning, and implementation. Additionally, collaborative programming can improve individual programming skills more than solo programming (Braught, Wahls, & Eby, 2011). In this research, we aim to extend our understanding of collaborative programming through the systematic integration of discourse and log analysis to determine how these collaborative factors impact group learning.

3. Methods

3.1 NetsBlox and RoboScape

NetsBlox, an extension of the Snap Programming language (http://snap.berkeley.edu/), is a visual programming environment created to teach students about distributed programming (Broll et al., 2017). NetsBlox allows projects to be shared among group members so that they can synchronously edit and execute various sections of their programs. This allows students to flow between different roles and maintain agency over their own work. Student actions are saved to a database as a particular action type, along with metadata such as the user id, project id, and Unix timestamp. This allows for the compilation of a comprehensive timeline of actions from a student and group perspective.

Communication with a physical robot is enabled through Remote Procedure Calls (RPC). One such RPC, RoboScape (Lédeczi et al., 2019), has a set of specific commands that students can use to communicate with Parallax robots nearly instantaneously. RoboScape, like Zero Robotics (Saenz-Otero et al., 2010) and others, uses a robotics-based framework as an entertaining and educational approach to encourage collaboration through competition. Students work together to strategize for each specific competition and troubleshoot errors in their programs, observing any changes in real-time.

3.2 Study Design and Analysis Framework

Thirty-eight high school students spread across two, week-long summer camps completed our intervention using the NetsBlox and RoboScape environment. Like Karaman et al. (2017), we emphasized robotics software over hardware while incorporating project-based learning and collaborative requirements within a module-based structure. Day one began with an introduction to the NetsBlox environment to establish a baseline of programming proficiency. Day two was the first collaborative day for students. They primarily created simple manual driving programs using RoboScape to increase familiarity with the system and grow comfortable working as a team. Next, they progressively studied more complex cybersecurity techniques, with groups implementing both attack and defense strategies. This culminated in a final competition that required both collaboration and programming skills to succeed while fending off cyber-attacks (Lédeczi et al., 2019; Yett et al., 2020a).

Table 1. *Our Analysis Framework*

	Learning	Individual Action Analysis	Sequence Analysis	CPS Discourse
Desired Characteristic Indicative of Shared Understanding	Learning gains experienced by all group members	Equal distribution of programming activities	Utilization of strategies identified as supporting program development	Predominantly interactive discourse for a shared understanding of the problem
Analysis – Entire Intervention	Average normalized change of results in CT and cybersecurity	Actions taken by each group member, Gini coefficient	Differential sequence mining of full sequence of actions	N/A
Analysis – Tug of War project	Average normalized change of results related to project	Actions taken by each group member, Gini coefficient	Markov chains of actions taken during the project	Application of framework to the discourse of students during the project

The students separated into dyads and triads, choosing their own groups. To be considered a group for this analysis, the students had to work together for at least three of the four collaborative days of the intervention. Each student also had to complete the pre- and post-tests. Twelve groups were included – six dyads and six triads - spread throughout the two weeks of the camp. The framing of our collaborative analysis approach is presented in Table 1 above.

Data logged during the intervention was queried from the NetsBlox database. We extracted the action data from the database and categorized them as Solution Construction (SC) and Solution Assessment (SA) actions (Kinnebrew et al., 2017). SC actions were subdivided into (1) *FillField* actions when students changed the values of variables, lists, or robot commands; (2) *AddBlock* or (3) *RemoveBlock* actions when students added a new block to their program or deleted an existing block; (4) *ConnectBlock* or (5) *DisconnectBlock* actions when students attached a block to existing blocks or removed a block from an existing sequence of blocks, respectively. SA actions were likewise split into (1) *StartSimulation* actions when students clicked on the "green flag" within the NetsBlox environment to begin running a full program or (2) *StartScript* actions when students executed only a small portion of their code. Additionally, *Idle* events occurred when students within a group had a gap of 160 seconds or more between two actions. This value was determined by the 95th percentile of the time intervals.

We applied collaborative discourse analysis (Snyder et al., 2019; Snyder, Hutchins, Biswas, & Grover, 2019) superimposing the ICAP (Interactive-Constructive-Active-Passive) framework of engagement modes (Chi & Wylie, 2014) onto the Weinberger & Fischer (2006) framework component of social modes during knowledge construction. This combined framework allows for analysis of students' engagement as well as their social modes. The resulting framework consists of eight components - we consider six of them within our analysis. (1) *Interactive conflicted-oriented consensus building* (ICO) occurs when most group members are participating in the model building and are disagreeing about what needs to be done. (2) *Interactive integration-oriented consensus building* (IIO) occurs when most group members are participating but no member is taking a strong position on what the next step should be. (3) *Interactive quick consensus building* (IQ) occurs when most group members are participating in the model building and one member's suggested contribution is accepted with little to no discussion. (4) *Interactive elicitation* (IE) occurs when most group members are participating in the model building and one student questions the group. (5) *Constructive externalization* (CEX) occurs when only one group member is participating in the model building and narrates their actions and thought processes. (6) *Constructive elicitation* (CEL) occurs when only one student is participating and asks the other non-participating group members a question. Inter-rater reliability was checked by two coders on a 36-utterance segment of discourse and resulted in a Cohen's kappa value of k = 0.73, indicating good agreement.

We answered RQ1 via the "Learning" and "Sequence Analysis" columns of our framework (Table 1). The pre- and post-tests (targeting CT and cybersecurity) were graded using normalized change (Marx & Cummings, 2007) to measure the growth from pre- to post-test for each individual student. These were combined to provide an average normalized change (ANC) based on the number of students in the group, leading to a median split into "Large Improvement" (LI) and "Small or No Improvement" (SNI) groups. We applied Differential Sequence Mining (DSM) to evaluate key strategies implemented by LI and SNI groups (Kinnebrew, Loretz, & Biswas, 2013; Kinnebrew et al., 2015). DSM compares the instance support, or i-support (I-S), of the compared groups. I-S is defined as the average number of times the pattern was used by each group.

A competition from day three of the intervention called "Tug of War" (ToW) was chosen for further analysis drawing from both the action logs and the discourse of students. In this competition, students fought for control over a robot. Each group would simultaneously send commands to a single robot, attempting to maneuver it towards opposing objectives. This project was chosen because of its highly collaborative nature, as teams of students had to program and strategize together in order to succeed. It was also chosen due to the relation between the sections of code often used by successful teams and two questions on the pre-post-tests related to loops and denial of service attacks. Students learned about and implemented denial of service attacks - and defenses for such attacks - in preparation for the ToW competition, while loops were a necessary component of these attacks.

To answer RQ2 within the ToW context, we required all four columns of the framework presented in Table 1. We selected two groups for analysis – one LI group and one SNI group. Log data was used to compute the Gini coefficient (Dorfman, 1979) as a metric to evaluate individual contributions to group performance. Gini coefficients indicate the share of actions taken by each student, with smaller values corresponding with more equally divided actions. We computed ANC on the specific ToW-related questions as well as determining the action counts (adjusted for number of group members and the half-day length of the project) and Gini coefficient during this time. We also generated Markov chain (MC) models (Russell & Norvig, 2003) with students' sequences of log actions to analyze overall group programming sequences. Each state in the state space is represented by an action which occurred at time t, and a transition probability indicates the likelihood for another action to occur at time t+1 with the fitted MC model. Finally, we manually applied our collaborative discourse framework to the discussions of the chosen groups during this project, resulting in counts from each category as well as specifically highlighted discussions related to the programming strategies of each group.

4. Results

Summative results indicated learning gains across CT and cybersecurity (Yett et al., 2020a). Further analysis revealed relationships between average normalized change and results from action log data, the strongest of which corresponded to solution construction actions (Yett et al., 2020b).

4.1 RQ1: *What programming behaviors as observed via differential sequence mining (DSM) distinguish groups of students with large improvements during the intervention versus those without?*

Using the DSM method, we found action sequence patterns that had statistically significant differences in the instance support (I-S) metric between "Large Improvement" (LI) and "Small or No Improvement" (SNI) groups. Table 2 lists the patterns and the DSM results ranked by the effect size of the differences. The threshold of the *s*-Support (percentage of groups supporting this pattern) of the DSM was set at the 80% level, and there was no gap allowed between actions.

Table 2. *DSM Results*

	Pattern	I-S (LI)	I-S (SNI)	p-value	Effect Size
1	ConnectBlock x 9	1.3	4.2	0.02	1.69
2	StartScript, DisconnectBlock, StartScript, RemoveBlock	1.2	0.2	0.02	1.65
3	StartSimulation, DisconnectBlock	6.0	16.7	0.03	1.62
4	StartSimulation x 6	4.0	17.8	0.04	1.54
5	FillField, StartScript, FillField, StartScript, ConnectBlock	1.8	0.7	0.03	1.49

SNI groups were more likely to perform a long series of edits to their models. The pattern *ConnectBlock* × *9* (pattern 1) appeared in 100% of the SNI groups, who also had much larger I-S of this pattern (4.2 instances per group). This depth-first programming approach as opposed to programming in parts has been reported in the literature (Grover, Bienkowski, Niekrasz, & Hauswirth, 2016). Secondly, SNI groups used more solution assessment (SA) actions overall, especially the *StartSimulation* action once or multiple times in succession (e.g. pattern 4). Though SA actions offer important feedback to construct models effectively, their excessive use implies running simulations without modifications to the models, which indicates an inability to comprehend the information provided by simulations and was linked to SNI students' overall poorer learning performance.

LI groups showed a more systematic programming approach. Patterns 2 and 5 both could indicate productive troubleshooting behaviors of students testing new blocks or variable values. This is further established by the fact that although SNI groups implemented pattern 3 more (which is similar to the initial sequence of pattern 2), the LI groups may have done so in a more systematic way, as they played the simulation, disconnected a block that may have contained an error, played again to test, and (presumably correct in their hypothesis) removed the block. We hypothesize that the increased use of pattern 3 by SNI groups was due to unsystematic debugging methods used to identify code errors, an indication of ineffective trial-and-error approaches (Murphy et al., 2008). Pattern 5 shows these LI groups changing variable values (*FillField*) followed by testing those changes depending on the current task at hand (*StartScript*) and adding the block they were modifying to the full program (*ConnectBlock*).

4.2 RQ2: *How did the relative presence or absence of collaboration during a specific project affect student understanding as indicated by learning gains?*

Our case study compares the discourse and log data of an LI group (Group 7) to a SNI group (Group 12) during the Tug of War (ToW) task.

4.2.1 Group 7

Group 7 (G7) was a dyad that worked together from Day 2 through Day 4. They were average in terms of Gini coefficient (0.19) and total actions taken per student per day (364), indicating that they evenly contributed to the projects. They also showed the most improvement among any groups with an average normalized change (ANC) of 0.92. Their results during ToW were similar to their overall scores, with an adjusted 321 actions, Gini coefficient of 0.19, and ANC of 1 (as each student began with missing at least one of the questions on the pre-test but each answered the questions correctly on the post-test).

G7 implemented a few sequences (Figure 1) that relate to our findings from Section 4.1. For instance, transitions such as *FillField* to *StartScript* may be associated with the productive troubleshooting behaviors described. They also frequently transitioned between *FillField* and *StartSimulation*, which indicated a debugging strategy by tweaking the parameters between tests. This model also indicates potentially new troubleshooting techniques, such as running an individual script for local testing followed by running the entire program. Overall, the group exhibited a variety of behaviors depending on the current needs of the project, with at least one non-recursive transition out of each state. One possible cause for concern is the high likelihood of *Idle* to *StartScript*, which could indicate a return to the program after some time spent off-task and an immediate test of a feature that had already been attempted unsuccessfully. However, this may also relate to discussion time and the running of the simulation for further verification. Additionally, students in this group had a 38% likelihood of transitioning from *AddBlock* to *RemoveBlock*. Though this is not necessarily indicative of mistakenly adding a block only to remove it, it could signify program changes without proper testing.

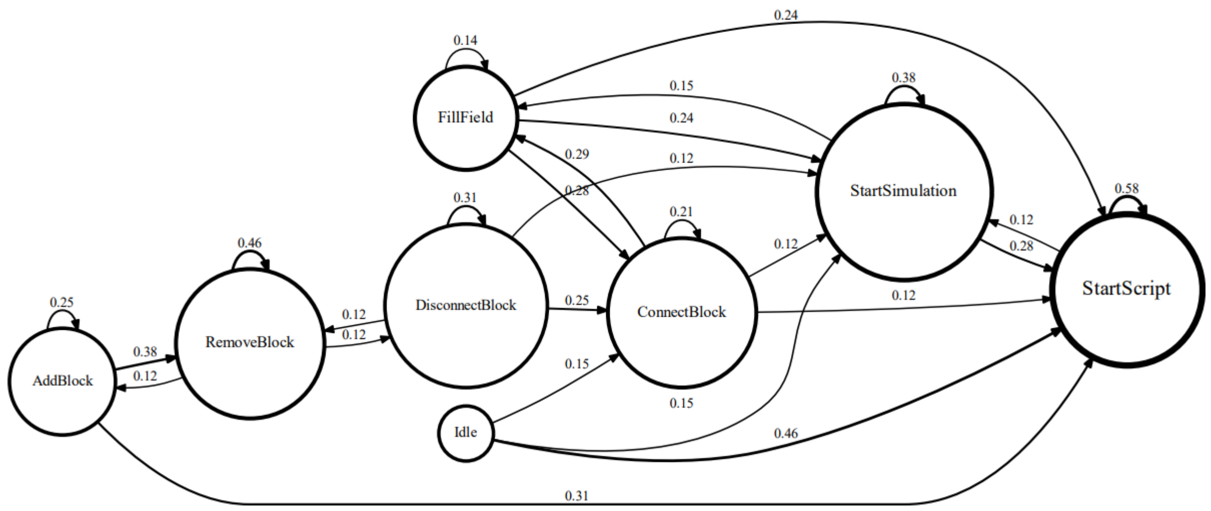

Figure 1. Markov Chain Results from Group 7

G7 demonstrated predominantly interactive discourse during programming strategy time segments (Table 3). Coded collaborative discourse indicated that only 32% of utterances coded were constructive, with the group's discourse targeting mainly *interactive integration-oriented consensus building* (IIO) and *interactive elicitation* (IE) in developing their models. In the provided example, group members used explanatory words such as "because" that indicate the group's ability to reason about their programming actions. This coincides with the literature in that group reasoning and explanatory process are supportive of developing a shared understanding (Fischer et al., 2002; Hmelo-Silver, 2003). Both students logically contributed approximately equally to all Interactive discussions, as interaction is required to achieve such a coding. However, G7-S2 otherwise dominated the conversation, spending more time in a Constructive state attempting to engage their partner while G7-S1 responded tersely or not at all. G7-S2 also commonly explained their programming and competition strategies to G7-S1 and several outsiders to the group, resulting in high quantities of *constructive externalization* (CEX).

Table 4. *Excerpt from Group 7 Highlighting their use of Programming Strategy*

Student	Discourse	CPS Code
S1	"my sent" isn't working	ICO
S1	what did you do?	ICO
S2	me?	ICO
S1	why are all these right there	ICO
S2	because they're for the sent commands	ICO
S1	why are they under forwards (custom blocks)	ICO
S2	they're under there so that when we send a command it changes "my sent". I put them there for a reason	ICO
S1	well "my sent" is going but the "sent" overall isn't changing	ICO
S2	no it does see, "change sent by 1" right there	ICO
S1	no, ok. that's not changing. neither is "my sent"	ICO
S2	where is our robot? there it is	IIO
S1	"sent" itself is not changing. right there. the overall "sent"	IIO
S2	mine is	IIO
S1	but "my sent" is changing	IIO

4.2.2 Group 12

Group 12 (G12) was a triad that worked together from Day 2 through Day 5. They were average in terms of Gini coefficient (0.20) but took the fewest actions (229) and were the only group to regress from pre- to post-test (ANC = -0.08). At an individual level, two students recorded ANC's of 0, while one recorded a -0.33 – an additional positive consideration is that the regressing student started with a perfect score and another student only missed one question on each of the pre- and post-tests. Though they took a similar number of actions during ToW after accounting for the length of the session (209), their actions were less evenly divided than usual (Gini = 0.32). Despite this, they saw some improvement in the ToW-related pre-post questions (ANC = 0.67), all from the growth of one student as the other two had answered both questions correctly on the pre-test.

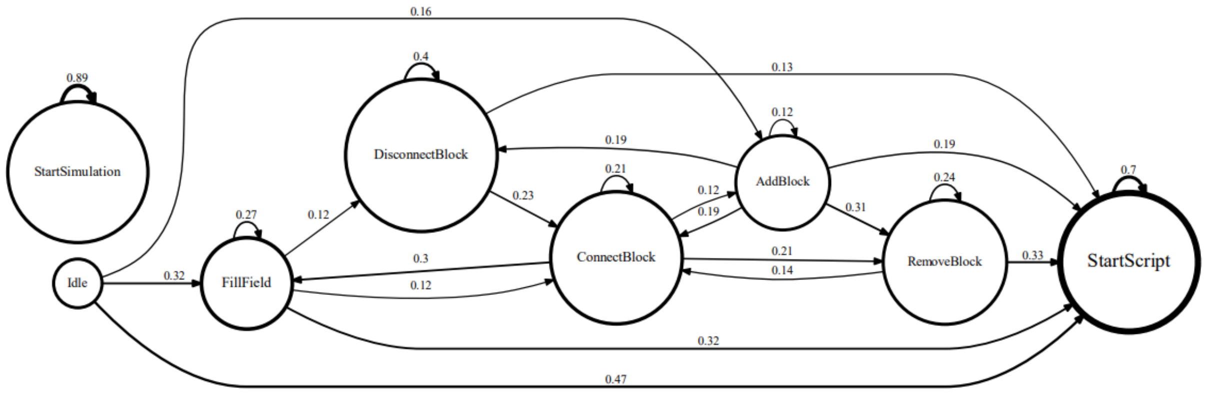

Figure 2. Markov Chain Results from Group 12

An immediate cause for concern upon examination of Figure 2 is the high likelihood of repeated *StartSimulation* (89%) and *StartScript* (70%) actions. This coincides with our finding in Section 4.1 demonstrating that SNI groups were likely to implement long sequences of SA actions. Some of this can be explained by the nature of the task, as students often needed to execute at least a few scripts (or the full simulation) in sequence in order to perform well. However, usage at this magnitude indicates repeated examination of the impact of one or a few changes without any modifications in between. This group displayed some positive attributes as well. For example, they rarely transitioned from *AddBlock* to *RemoveBlock* or *RemoveBlock* to *RemoveBlock*, instead showing the desired behavior of *StartScript* following these changes to their program. On the other hand, the model edit actions and the simulation actions were disjoint: unlike that of Group 7, the MC model of this group had no connection between *StartSimulation* to any model edit action, and the *ConnectBlock* to *StartScript* link that is an important part of a debugging strategy was missing. As a result of this disjoint modeling and testing behavior, this group of students did not seem to be able to build and modify the ToW model successfully.

G12's discourse (Table 4) is defined by significant use of constructive discourse with little to no explanations for programming changes. For instance, S1 says "as long as you use that to move then we shouldn't get locked out," with no reasoning. Overall, 45% of the group's coded utterances were constructive, compared to G7's 32%. Of the interactive discussions, the group implemented mainly IE, in which one student questions the group. G12-S1 exhibited similar tendencies to G7-S2, taking on a leadership role and generating more discussion than their peers. Likewise, G12-S2 was similar to G7-S1, still contributing to the programming process but not as frequently. One group member, S3, rarely participated in discussions (accounting for less than 10% of the group's coded utterances). Overall Gini data for the group does indicate that all students normally participated somewhat evenly, so this could just be an outlier caused by external factors upon G12-S3.

Table 4. *Excerpt from Group 12 Highlighting their use of Programming Strategy*

Student	Discourse	CPS Code
S1	as long as you use that to move then we shouldn't get locked out	ICO
S2	It's not working	ICO
S1	Okay [name]	CEX
S1	so	CEX
S1	I set this to number 2	CEX
S2	okay	CEX
S1	So when I press 2	CEX
S2	Press 2	CEX
S1	Actually wait we don't need number 2 that won't work. We need the flag	CEX
S2	The flag?	CEL
S1	And it needs to be a forever loop	CEL

5. Discussion and Conclusion

At an overall level, a few characteristics stand out as being common between the two groups. They spent an approximately equivalent amount of their conversation time on *interactive conflicted-oriented consensus building* (ICO) (26 for G7 and 23 for G12). Though this behavior is argumentative in nature, at these low levels it can often be beneficial in terms of an improved shared understanding and moving the project forward (Weinberger & Fischer, 2006). The two groups also displayed a similar quantity of each kind of constructive discourse and an almost complete lack of interactive quick consensus building. On the other hand, G7 conducted almost twice as many IE exchanges than their counterparts in G12 (110 to 66) along with almost four times as many cases of IIO (81 to 22). Overall, G7 displayed almost entirely positive features from the framework established in Table 1, learning together and showing their collaborative skills through Individual Action Analysis, Sequence Analysis, and Collaborative Problem-Solving (CPS) Discourse Analysis. On the other hand, despite some promising trends in Sequence Analysis and Learning during Tug of War (ToW), G12 appeared to struggle to work together and reach the desired level of achievement.

The combination of NetsBlox and RoboScape provides a convenient venue for studying collaborative programming. We contributed new findings in the analysis of programming actions taken by students while collaborating, considering both the distribution of actions amongst group members and the full combined sequence of actions of the group. We also used the available data to highlight two groups as they interacted during a collaborative task. Each of these groups showed Interactive behaviors and students filling specific collaborative roles even without the assignment of predefined roles. The dyad was more successful in terms of sharing actions and showing improvements from pre-post, while the triad seemed to leave one student behind and struggle more. Taken together, these groups indicate that collaboration on the ToW project could have some impact on pre-post results. However, a more complete dataset would be necessary to validate this claim after addressing a few shortcomings.

One limitation of the system is that network issues can lead to interruptions in collaboration as projects become desynchronized; an integrity check to ensure consistency between the distributed applications could help mitigate this issue. We also allowed groups to choose their own members, possibly skewing results as friends teamed up and worked well together while students that did not know each other had to build their relationship. To somewhat mitigate this disadvantage, lectures on collaboration (Karaman et al., 2017) would be a welcome addition. A final area for improvement is in collecting audio/visual data throughout the intervention, as problems such as microphones being too far away from students and students manually canceling recordings caused us to lose some or all data for certain groups. The combination of these and other factors forced the decision to compare a dyad and triad for our case studies. Ideally, we would compare only dyads or only triads for consistency. We plan to remedy the discussed limitations and introduce additional analysis methods in order to progress the understanding of the effectiveness of our platform.

Acknowledgements

This material is based in part upon work supported by National Security Agency Science of Security Lablet H98230-18-D-0010 and National Science Foundation grants CNS-1644848, CNS-1521617, and DRL-1640199. Any opinions, findings, and conclusions expressed in this material are those of the author(s) and do not necessarily reflect the views of the US Government.

References

Branch, J. L. (2013). The Trouble with Think Alouds: Generating Data using Concurrent Verbal Protocols. In *Proceedings of the Annual Conference of CAIS*.

Braught, G., MacCormick, J., & Wahls, T. (2010, March). The Benefits of Pairing by Ability. In *Proceedings of the 41st ACM Technical Symposium on Computer Science Education* (pp. 249-253).

Braught, G., Wahls, T., & Eby, L. M. (2011). The case for pair programming in the computer science classroom. ACM Transactions on Computing Education (TOCE), *11*(1), 1-21.

Broll, B., Lédeczi, Á., Volgyesi, P., Sallai, J., Maróti, M., Carrillo, A., ... & Lu, M. (2017, March). A Visual Programming Environment for Learning Distributed Programming. In *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education* (pp. 81-86).

Chi, M. T. H., & Wylie, R. (2014). The ICAP Framework: Linking Cognitive Engagement to Active Learning Outcomes. *Educational Psychologist*, *49*(4), 219-243.

College Board. AP Computer Science Principles Curriculum Framework. 2020.

Dorfman, R. (1979). A Formula for the Gini Coefficient. *The Review of Economics and Statistics*, 146-149.

Fischer, F., Bruhn, J., Gräsel, C., & Mandl, H. (2002). Fostering collaborative knowledge construction with visualization tools. *Learning and Instruction*, *12*(2), 213-232.

Garrison, D. R., & Akyol, Z. (2013). The Community of Inquiry Theoretical Framework. In *Handbook of Distance Education 3* (pp. 104-120).

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*(6), 401-412.

Grover, S., Bienkowski, M., Niekrasz, J., & Hauswirth, M. (2016, April). Assessing Problem-Solving Process at Scale. In *Proceedings of the Third (2016) ACM Conference on Learning @ Scale* (pp. 245-248).

Grover, S., Bienkowski, M., Tamrakar, A., Siddiquie, B., Salter, D., & Divakaran, A. (2016, April). Multimodal Analytics to Study Collaborative Problem Solving in Pair Programming. In *Proceedings of the Sixth International Conference on Learning Analytics & Knowledge* (pp. 516-517).

Hmelo-Silver, C. E. (2003). Analyzing collaborative knowledge construction: Multiple methods for integrated understanding. *Computers & Education*, *41*(4), 397-420.

Karaman, S., Anders, A., Boulet, M., Connor, J., Gregson, K., Guerra, W., ... & Vivilecchia, J. (2017, March). Project-Based, Collaborative, Algorithmic Robotics for High School Students: Programming Self-Driving Race Cars at MIT. In *2017 IEEE Integrated STEM Education Conference (ISEC)* (pp. 195-203). IEEE.

Kinnebrew, J. S., Loretz, K. M., & Biswas, G. (2013). A Contextualized, Differential Sequence Mining Method to Derive Students' Learning Behavior Patterns. *JEDM | Journal of Educational Data Mining*, *5*(1), 190-219. https://doi.org/10.5281/zenodo.3554617

Kinnebrew, J. S., Segedy, J. R., & Biswas, G. (2017). Integrating Model-Driven and Data-Driven Techniques for Analyzing Learning Behaviors in Open-Ended Learning Environments. *IEEE Transactions on Learning Technologies*, *10*(2), 140-153.

Lédeczi, Á., Maróti, M., Zare, H., Yett, B., Hutchins, N., Broll, B., ... & Koutsoukos, X. (2019, February). Teaching Cybersecurity with Networked Robots. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education* (pp. 885-891).

Marx, J. D., & Cummings, K. (2007). Normalized Change. *American Journal of Physics*, *75*(1), 87-91.

Murphy, L., Lewandowski, G., McCauley, R., Simon, B., Thomas, L., & Zander, C. (2008). Debugging: the Good, the Bad, and the Quirky--a Qualitative Analysis of Novices' Strategies. *ACM SIGCSE Bulletin*, *40*(1), 163-167.

Noroozi, O., Alikhani, I., Järvelä, S., Kirschner, P. A., Juuso, I., & Seppänen, T. (2019). Multimodal data to design visual learning analytics for understanding regulation of learning. *Computers in Human Behavior*, *100*, 298-304.

Rogat, T. K., Linnenbrink-García, L., & DiDonato, N. (2013). Motivation in Collaborative Groups. *International Handbook of Collaborative Learning*, 250-267.

Roschelle, J., & Teasley, S. D. (1995). The Construction of Shared Knowledge in Collaborative Problem Solving. In *Computer Supported Collaborative Learning* (pp. 69-97). Springer, Berlin, Heidelberg.

Russell, S. J. and Norvig, P. (2002). Artificial Intelligence - A Modern Approach, 2nd Edition. *Prentice Hall Series in Artificial Intelligence*. Prentice Hall.

Saenz-Otero, A., Katz, J., Mohan, S., Miller, D. W., & Chamitoff, G. E. (2010, March). ZERO-Robotics: A Student Competition Aboard the International Space Station. In *2010 IEEE Aerospace Conference* (pp. 1-11). IEEE.

Shi, J., Shah, A., Hedman, G., & O'Rourke, E. (2019, May). Pyrus: Designing A Collaborative Programming Game to Promote Problem Solving Behaviors. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (pp. 1-12).

Snap!: a Visual, Drag-and-Drop Programming Language. (n.d.). Retrieved from http://snap.berkeley.edu/snapsource/snap.html.

Snyder, C., Hutchins, N., Biswas, G., Emara, M., Grover, S., & Conlin, L. (2019, June). Analyzing Students' Synergistic Learning Processes in Physics and CT by Collaborative Discourse Analysis. In *13th International Conference on Computer-Supported Collaborative Learning* (pp. 360-367).

Snyder, C., Hutchins, N., Biswas, G., & Grover, S. (2019, June). Understanding Students' Model Building Strategies Through Discourse Analysis. In *AIED 2019: International Conference on Artificial Intelligence in Education* (pp. 263-268). Springer, Cham.

Thompson, J. D., Herman, G. L., Scheponik, T., Oliva, L., Sherman, A., Golaszewski, E., ... & Patsourakos, K. (2018). Student Misconceptions about Cybersecurity Concepts: Analysis of Think-Aloud Interviews. *Journal of Cybersecurity Education, Research and Practice*, *2018*(1), 5.

Tsan, J., Lynch, C. F., & Boyer, K. E. (2018). "Alright, What do we Need?": A Study of Young Coders' Collaborative Dialogue. *International Journal of Child-Computer Interaction*, *17*, 61-71.

Vrzakova, H., Amon, M. J., Stewart, A., Duran, N. D., & D'Mello, S. K. (2020, March). Focused or Stuck Together: Multimodal Patterns Reveal Triads' Performance in Collaborative Problem Solving. In *Proceedings of the Tenth International Conference on Learning Analytics & Knowledge* (pp. 295-304).

Weinberger, A., & Fischer, F. (2006). A Framework to Analyze Argumentative Knowledge Construction in Computer-Supported Collaborative Learning. *Computers & Education*, *46*(1), 71-95.

Werner, L., Denner, J., Campe, S., & Kawamoto, D. C. (2012, February). The Fairy Performance Assessment: Measuring Computational Thinking in Middle School. In *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education* (pp. 215-220).

Werner, L., McDowell, C., & Denner, J. (2013). A First Step in Learning Analytics: Pre-Processing Low-Level Alice Logging Data of Middle School Students. *Journal of Educational Data Mining (JEDM)*, *5*(2), 11-37.

Yett, B., Hutchins, N., Stein, G., Zare, H., Snyder, C., Biswas, G., ... & Lédeczi, Á. (2020, February). A Hands-On Cybersecurity Curriculum Using a Robotics Platform. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education* (pp. 1040-1046).

Yett, B., Hutchins, N., Snyder, C., Zhang, N., Mishra, S., & Biswas, G. (2020, July). Evaluating Student Learning in a Synchronous, Collaborative Programming Environment Through Log-Based Analysis of Projects. In *International Conference on Artificial Intelligence in Education* (pp. 352-357). Springer, Cham.

Zakaria, Z., Boulden, D., Vandenberg, J., Tsan, J., Lynch, C., Wiebe, E., & Boyer, K. (2019, June). Collaborative Talk Across Two Pair-Programming Configurations. In *Proceedings of the 13th International Conference on Computer Supported Collaborative Learning (CSCL)* (pp. 224-231).

Zimmerman, B. J., & Pons, M. M. (1986). Development of a Structured Interview for Assessing Student Use of Self-Regulated Learning Strategies. *American Educational Research Journal*, *23*(4), 614-628.