Identifying Collaborative Problem-Solving Behaviors Using Sequential Pattern Mining

Yiqiu Zhou, University of Illinois, Urbana-Champaign
Qianhui Liu, University of Illinois, Urbana-Champaign

Qianhui (Sophie) Liu is a PhD student in the Department of Curriculum & Instruction, College of Education at UIUC. Her research interests are learning analytics, educational data mining, computer science education, and explainable AI.

Sophia Yang, University of Illinois, Urbana-Champaign

Sophia Yang is a second-year Ph.D. candidate with research work focused in the areas of Computing Education, Database Systems, Bioinformatics algorithms, Human-Computer Interaction, and interesting intersections of the above. Current research work focuses on quantitatively and qualitatively studying how students learn SQL by utilizing Bioinformatics alignment algorithms, aiming to create instructor and student-facing tools. Her work aims to better improve students' learning and instructors' teaching experiences in large-scale database courses.

Dr. Abdussalam Alawini, University of Illinois, Urbana-Champaign

Abdussalam Alawini is a Teaching Assistant Professor in the Department of Computer Science at the University of Illinois at Urbana-Champaign. He holds a bachelor's degree from the University of Tripoli and master's and doctoral degrees in Computer Science from Portland State University. Dr. Alawini has worked in various roles in the tech industry, including as a database administrator, lead software developer, and IT Manager. He conducts research on data management systems and computing education. Dr. Alawini is passionate about building data-driven, AI-based systems for improving teaching and learning.

Identifying Collaborative Problem-Solving Behaviors Using Sequential Pattern Mining

Abstract

With the increasing adoption of collaborative learning approaches, instructors must understand students' problem-solving approaches during collaborative activities to better design their class. Among the multiple ways to reveal collaborative problem-solving processes, temporal submission patterns is one that is more scalable and generalizable in Computer Science education. In this paper, we provide a temporal analysis of a large dataset of students' submissions to collaborative learning assignments in an upper-level database course offered at a large public university. The log data was collected from an online assessment and learning system, containing the timestamps of each student's submissions to a problem on the collaborative assignment. Each submission was labeled as quick (Q), medium (M), or slow (S) based on its duration and whether it was shorter or longer than the 25th and 75th percentile. Sequential compacting and mining techniques were employed to identify pairs of transitions highly associated with one another. This preliminary research sheds light on the recurring submission patterns derived from the amount of time spent on each problem, warranting further examination on these patterns to unpack collaborative problem-solving behaviors. Our study demonstrates the potential of temporal analysis to identify meaningful problem-solving patterns based on log traces, which may help flag key moments and alert instructors to provide in-time scaffolding when students work on group assignments.

Introduction

Collaborative learning, an educational approach involving groups of learners working together on a problem, has shown to increase productivity, achievement, and positive psychological and social outcomes[1]. Computer-Supported Collaborative Learning (CSCL), grounded in the Social Constructivism Theory, leverages technologies to facilitate and encourage interactions among students across domains[2]. Although CSCL has been incorporated into education by various studies[3,4,5,6], teachers and policymakers may lack understanding of how group collaboration can be effectively integrated into instructional strategies[7]. The use of CSCL technologies, pedagogies, and curricula by both teachers and students requires further investigation.

Past CS education research has attempted to detect individual-level problem-solving behaviors to assist struggling students, including identifying error-fixing patterns[8] and latent student profiles[9]. As more instructors adopt CSCL[7], a comprehensive understanding of collaborative processes is crucial to avoid unintended negative outcomes, including increased dropout rates and unmotivated students in introductory programming courses[10]. Our study addresses this issue by investigating

problem-solving temporal behaviors in a collaborative setting to aid instructors in designing CSCL courses that account for individual and collaborative learning strategies, potentially improving students' engagement in group roles and activities[11].

In this paper, we study collaborative problem-solving strategies in a large-enrollment database course offered at University of Illinois Urbana-Champaign. This course teaches three database query languages, including Structured Query Language (SQL, a query language for relational databases), MongoDB (a document-oriented database), and Cypher (the query language for Neo4J, a graph database system). The course offers in-class collaborative programming activities and uses an online assessment learning system PrairieLearn[12]. We extracted submission log data from collaborative assessments, including student ID (students' sensitive information has been omitted as per our IRB guidelines), submission timestamps, and submission grades (correct/incorrect). Based on our temporal analysis of this log data, we aim to address the following research questions:

RQ1: What recurring temporal patterns can be identified in the sequence of student submission attempts on SQL, MongoDB and Neo4J query languages? To answer RQ1, each submission was labeled as Q, M, or S based on whether its duration was shorter or longer than the 25th and 75th percentile. We then applied sequence compacting techniques to reduce the length of sequences in a more meaningful way related to problem-solving across the three database query languages.

RQ2: How can these patterns provide insights into group problem-solving strategies? To answer RQ2, we conducted sequential pattern mining, clustering, and correlation analysis to identify latent patterns that characterize various problem-solving strategies.

By answering these research questions, this study aims to explore the potential of temporal information, specifically the time spent on each submission attempt, in revealing recurring patterns in group submission sequences which could offer insights into how groups approach and solve the assignments. The paper is structured as follows: related work covers CSCL literature on temporal analysis, methodology details the data source and analysis pipeline, and the remaining sections present the results, implications, future work, and limitations of the study.

Related Work

Collaborative learning through CSCL has been widely utilized and researched in introductory programming education to promote cognitive strategies, logical thinking, and engagement through social interactions[13]. Several collaborative resources and techniques, such as collaborative programming editors[6], scaffolds for peer review and problem discussion[4], group chat[5], and CATME tools for team formation and evaluation[14], have been developed to support CSCL. Furthermore, Learning Analytics (LA), which involves analyzing data about learners to understand and enhance the learning process and environments[15], has the potential to unlock valuable information about collaborative learning behaviors. Yet, limited research has been conducted on the impact of LA in improving collaborative coding assignments[16]. Some previous studies, such as those by Kompan & Bielikova[17], Berland et al.[18], and Lu et al.[19], explored the use of LA to enhance teamwork quality, identify at-risk students, and provide timely interventions in collaborative programming activities. However, these studies mainly focused on analyzing

single-student learning behaviors. In contrast, our study examines group-level problem-solving behaviors by analyzing submissions made by the entire group to collaborative assignments.

Temporal analysis is an LA approach that uses Social Network Analysis (SNA), sequential analysis, visualizations, statistical discourse analysis, and epistemic network analysis to uncover short- and long-term patterns in CSCL[13]. For instance, SNA is used to detect direct communication in collaborative discussion posts[20], and sequential analysis is applied to knowledge-building discourse to detect productive patterns[21]. Our study is the first to analyze temporal patterns in group submissions to collaborative learning assignments, aiming to identify collaborative problem-solving strategies and investigate behaviors across several query languages. In our analysis, we also used a unique sequence compacting method to remove repetitive labels and simplify frequent patterns. This technique has been applied in other diverse fields[22] and this novel use in detecting learners' behaviors can present the opportunity to explore its potential in education. We collected and analyzed log data in three database query languages (SQL, MongoDB, and Cypher) to identify CSCL patterns that can help instructors design effective collaborative activities for various topics.

Methodology

In this section, we outline our data sources and discuss methods for generating temporal patterns, which involves labeling submissions, extracting patterns, and examining their correlations.

Data Source

Data was collected from 1,322 students enrolled in an upper-level undergraduate and graduate database course across three semesters (Fall 2020, Spring 2021, Fall 2021). This course adopted a flipped-classroom format with pre-recorded lectures and collaborative in-class assignments with three to five exercises on PrairieLearn. This learning platform allows unlimited resubmissions with immediate feedback and discourages round-robin strategies. The auto-grading system only awards full credit for reasonable effort. Our study utilized timestamp and submission data while adhering to IRB protocols and safeguarding student privacy through anonymization.

Pattern Generation

This exploratory study aims to discover temporal patterns that illuminate group problem-solving behaviors. It is important to emphasize that our analysis is conducted at the group level since students submit assignments and receive credits collectively. As a result, all log traces within the same group are aggregated to derive group-level submission patterns. Specifically, we focus on patterns derived from the time spent on each submission attempt, employing sequential pattern mining techniques to identify patterns potentially reflecting group problem-solving strategies. Our analytical pipeline comprises the following steps:

1. Submission Label

PrairieLearn platform supports two types of saving events: students can either save current progress for later continuation or save as a submission attempt for automatic grading. The 'save

and grade' event was filtered to extract submission attempts for grading. This event log allowed us to evaluate the time spent on each submission attempt by computing the time interval between two consecutive 'save and grade' events for the same problem. The submission attempts were then classified into three categories: quick (Q), medium (M), and slow (S), based on the 25th and 75th percentile of submission time. Specifically, submissions with a duration shorter than the 25th percentile of all submission times were classified as quick (Q), those that were within the 25th and 75th percentile were labeled as medium (M), and those longer than 75% of all submission times were classified as slow (S). In order to account for the potential discrepancies between different query languages and course structures across semesters, this classification process was conducted separately for each dataset per query language and semester.

2. Pattern Extraction

Since it would be difficult to conclude any useful patterns that return meaningful interpretations on group problem-solving behaviors from such fine-grained labeling, we developed a sequence compacting algorithm to extract patterns from the labeled submission sequences. This sequence compacting algorithm allows us to reduce the size of the labeled submission sequences while preserving the meaningful patterns of problem-solving behaviors among different query languages. Additionally, it is designed to capture consecutive repetitive patterns. which provides valuable insight into the dynamics of group problem-solving as well as common strategies for certain types of problems.

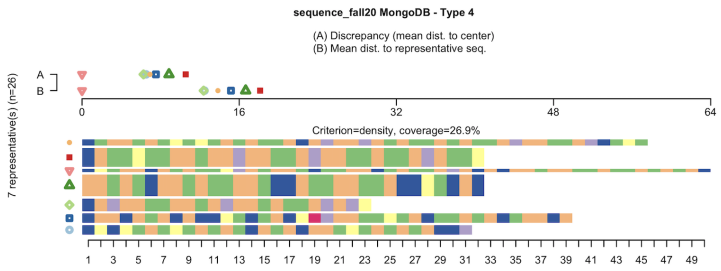
We, therefore, applied two compacting rules to the original sequences. First, three or more consecutive submissions with the same label were combined into one, under the assumption that three same consecutive labels were unlikely to occur by chance and may indicate recurrent use of certain problem-solving strategies. The second rule involved compacting consecutive Q and M substrings, given this was the most frequent substring found in the original submission sequences. As a result, seven patterns were generated by the rules introduced above: Q, Q.LNG, M, M.LNG, QM.LNG, S, and S.LNG. Specifically, Q stands for a single quick submission of the question, Q.LNG stands for a sequence of at least three quick submissions (e.g., Q-Q-Q), M stands for a single medium submission, M.LNG stands for a sequence of at least three consecutive medium submissions, QM.LNG stands for a sequence of multiple quick and medium submissions (e.g., Q-M-Q, M-Q-M), S stands for a single slow submission, and S.LNG stands for at least three consecutive slow submissions.

3. Pattern Analysis

We used agglomerative hierarchical clustering based on Ward's method[23] to analyze the data and categorize behavioral sequences. This approach group objects into clusters based on their similarity, sequentially merging the most similar clusters determined by variance. However, it was challenging to interpret the sequence clusters due to their complexity, particularly for those with lengths greater than 10, as shown in figure 1. Moreover, determining sequence clusters heavily impacted the sequence lengths, making it difficult to distinguish meaningful differences in patterns across sequences of varying lengths.

Fixed-length pattern compacting algorithm, such as the Variational Autoencoder[24], is a promising

Figure 1: An Example of Representative Sequences of Agglomerative Hierarchical Clustering. Sequences are plotted according to their representativeness, with bar height corresponding to the number of sequences assigned to them. Two parallel series of symbols at the top, each indicative of a representative submission sequence, are presented horizontally along a scale of theoretical distance.



approach for handling the variable pattern length issue. However, such algorithms often involve a significant amount of data transformation and loss of information, making it difficult to interpret the results. For this reason, our analysis aimed at understanding the relationships and influences between temporal patterns of collaborative submissions by focusing on their linkages. We used the L* metric to compare transition probabilities between seven submission patterns, which account for differences in base rates and are suitable for impossible transitions (e.g. Q $\rightarrow$ Q.LNG, S.LNG $\rightarrow$ S.LNG)[25]. A high transition probability between two patterns does not necessarily indicate the above chance as some patterns may occur more frequently in the sequence, leading to a higher frequency of specific transitions involving these patterns. L* metric compares the original transition probabilities with chance level, avoiding being confounded with the base rates of these submission patterns. L* illustrates the likelihood of transition between two patterns compared to a randomly ordered sequence, with a range of (-∞, 1], where the negative value represents the specific transition that is less likely to occur compared to the chance level, and 0 means this transition occurs as often as expected in a randomly ordered sequence.

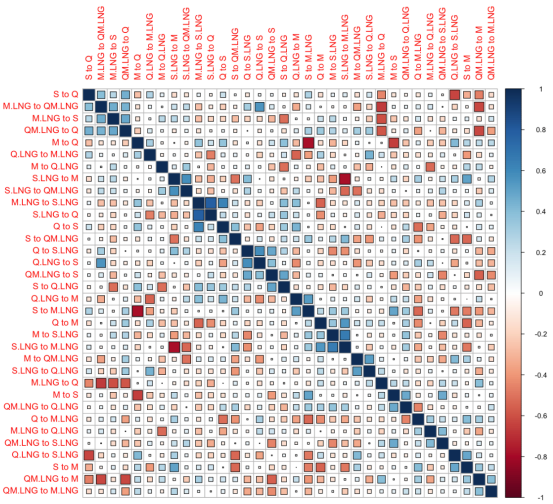
4. Correlation Analysis

We conducted a correlation analysis to assess the relationship between the L* metric values of temporal pattern transitions. As the metric indicates the probability of occurrence, this analysis enabled us to further investigate the patterns having a higher or lower likelihood of co-occurring together during the collaborative problem-solving process. For example, patterns that are positively associated have a higher likelihood of occurring together during the collaborative process. We present the results of this analysis in the next section.

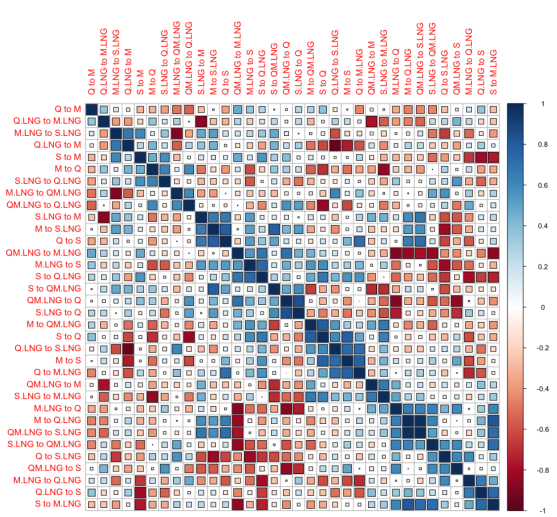
Results

Through the utilization of a correlation analysis based on the transition probability metric, we were able to identify multiple pairs of transitions which were highly associated with one another. This result reveals how certain transition pairs are more or less likely to be used in combination and observed together by students while they work on programming assignments. This provides valuable insight into the problem-solving strategies of groups and can potentially help teachers and researchers understand how students approach database programming tasks.

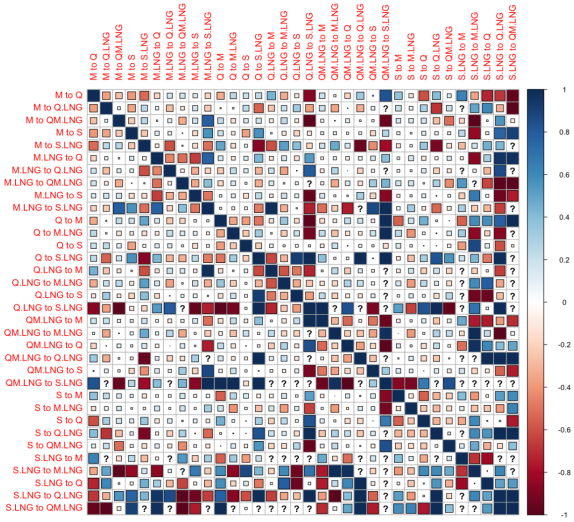
Figure 2: Matrix of the Correlation Transition Probabilities of SQL, MongoDB, and Neo4j; sizes of the squares represent the absolute value of the correlation coefficient; ? represents missing data



(a) SQL



(b) MongoDB



(c) Neo4j

SQL

The correlation matrix is presented in Figure 2a. The Pearson correlation analysis of the SQL dataset indicated a significant positive correlation between the occurrence probabilities of two transitions: transition from ***S to M.LNG*** and the transition from Q.LNG to M (r=0.524, p=0.021). This suggests an association between these two transitions, such that demonstrating the transition from ***S to M.LNG*** is more likely to coincide with demonstrating the transition from Q.LNG to M within the same group. Notably, the correlation matrix observed a significant negative correlation between the transition from ***S to M.LNG*** and the transition from Q.LNG to S.LNG (r=-0.48, p=0.037), suggesting an inverse relationship. It is noteworthy that the transition from ***S to M.LNG*** is associated with a different set of transition patterns starting from Q.LNG. Groups that demonstrate a higher frequency of ***S.LNG to M.LNG*** are more likely to demonstrate a transition from Q.LNG to M.LNG, but less likely to demonstrate a transition from Q.LNG to S.LNG.

Consecutive quick attempts (i.e., Q.LNG) can indicate a trial-and-error process or low-level mistakes fixation such as grammar mistakes, warranting further examination of the temporal pattern that follows Q.LNG in order to better interpret the group's problem-solving strategy. Consecutive slow attempts following Q.LNG (i.e., Q.LNG to S.LNG) may demonstrate a period of struggle where trial-and-error fails to yield progress in problem solving, resulting in extended time spent on each attempt. Alternatively, medium-length attempts (i.e., Q.LNG to M) may signify a more successful problem-solving occasion. Based on these findings, we further hypothesize that ***S to M.LNG*** represents a more desirable problem-solving pattern as it is positively associated with a relatively successful problem-solving occasion and negatively associated with a struggling occasion in programming problem solving.

Furthermore, the investigation into the association between the transition from M.LNG to S.LNG and the other two transitions from Q showed a statistically significant positive correlation between Q to S (r=0.621, p=0.005) and a statistically significant negative correlation between Q to M (r=0.548, p=0.015). This suggests that groups using successful SQL problem-solving strategies are more likely to transition from a slow attempt to a medium-length attempt, while those who encounter difficulties are more likely to transition from a medium-length attempt to a slow attempt.

MongoDB

The Pearson correlation analysis of the MongoDB dataset in Figure 2b revealed a significant positive relationship between the transition from M to Q.LNG and the transition from M to S.LNG (r = 0.967, p= 0.002). This indicates that groups who were more likely to make quick attempts (i.e., Q.LNG) were also more likely to enter into struggling moments, as reflected in their consecutive attempts with extended time (i.e., S.LNG). This result provides evidence of the strong correlation between two different problem-solving occasions in MongoDB which may be utilized to inform the development of automatic struggle detection and to identify learners who are in need of additional support.

Neo4j

The Neo4j correlation analysis illustrated in Figure 2c revealed a positive correlation between pairs of transitions containing the same element in different transition positions. For instance, the correlation between Q.LNG to M.LNG and M.LNG to Q (r=0.463, p=0.035) enabled the formation of a three-element transition, Q.LNG to M.LNG to Q. Additionally, the correlation between Q to M.LNG and M.LNG to S (r=0.440, p=0.046) suggested a three-element transition, Q to M.LNG to S. This finding has identified several problem-solving patterns in Neo4j that are commonly observed, which necessitates further in-depth analysis to comprehend these patterns and the implications of collaborative problem-solving strategies.

This section examined the relationship between transition probabilities and longer patterns in collaborative problem-solving processes across different query languages using log traces. The analysis revealed that certain transitions were associated with each other, such as the desirable pattern of ***S to M.LNG*** in SQL and longer transitions in Neo4j. MongoDB groups commonly used a trial-and-error approach and often encountered difficulties. These findings support

previous research that highlights the value of low-level log trace data, including time spent on submission attempts, in gaining meaningful insights.

Discussion

After applying compacting rules, we generated seven pattern types (Q, Q.LNG, M, M.LNG, QM.LNG, S, and S.LNG) in our study. We hypothesize that S and S.LNG (consecutive slow submissions) indicate a challenge that requires more guidance or clarification. This may be due to unproductive discussions or students feeling inept with the concepts being taught. We suggest instructors pay attention to groups with the S.LNG sequence in their submission pattern.

On the other hand, Q.LNG may indicate fruitful discussions among students, but it may also suggest a trial-and-error approach, which is not necessarily concerning. However, a high number of submission attempts along with Q.LNG may signal frustration and difficulty grasping query language concepts, indicating the need for instructors to monitor such groups closely.

Furthermore, we discovered that the transition probabilities of pairs are not transferable between the different query languages and that students exhibited different submission sequence patterns for different query languages. For example, we saw a statistically significant relationship between M.LNG to S.LNG and S.LNG to Q, forming the sequence M.LNG to S.LNG to Q for SQL in Neo4j, while there was little to no correlation among these pairs for MongoDB and Neo4j. We also saw there was a strong positive correlation between M to S.LNG and S.LNG to M.LNG for SQL, which had little to no correlation for MongoDB and Neo4j. We also observed that the relationships between transition pairs were more pronounced (stronger positives and stronger negatives) for MongoDB and Neo4j, compared to SQL (see figures 2a, 2b, and 2c). One possible explanation for these differences is that the inherent complexity or syntax variations in the query languages may lead to variations in submission sequences. Students might find it more or less challenging to grasp certain query languages based on their prior experience, which could influence the temporal patterns observed. Furthermore, the nature of the problems posed to the students and the optimal approaches for addressing them in each query language could contribute to the differences in submission patterns. Given these factors, it is crucial to consider the unique features and potential learning curves associated with each query language when interpreting the identified patterns and their implications. Consequently, we advise against instructors extrapolating findings from these transition pairs to apply to other query languages for which they were not computed.

Limitations and Future Work

This exploratory study uncovered temporal patterns in students' submission sequences across several query languages but has some limitations; although significant relationships were found between transition pairs for each analyzed query language, they may not apply to other query languages. Furthermore, our data is from one institution, and more research across multiple institutions is needed. Additionally, more pronounced relationships were observed among the MongoDB and Neo4j query languages' transition probabilities without explanation.

To better understand the reasons behind certain query languages having more pronounced

relationships and to validate our interpretations, we propose conducting a qualitative study by interviewing students. We plan to couple current exploratory study with performance data and submitted queries to more strongly identify successful problem-solving patterns and groups of students in need of support. Ultimately, we aim to develop an automated analysis tool for instructors to identify students who may require help during lecture sessions or collaborative assignments, as some students who would benefit from help may feel shy from seeking it.

Conclusions

Our study seeks to address the lack of knowledge about student group problem-solving behaviors in a collaborative learning setting and assist instructors in understanding the temporal patterns in student group submission sequences. Applying two compacting rules, we identified seven submission patterns - Q, Q.LNG, M, M.LNG, QM.LNG, S, and S.LNG. We found correlations between transition probabilities for each of the SQL, MongoDB, and Neo4j query languages, which were specific to each language and did not generalize to other languages. These temporal patterns can detect recurring trends in student problem-solving behaviors and provide instructors with insights into group dynamics and the need for their intervention. Our future work will involve incorporating student query submission and performance data to gain a better understanding of cognitive processes in collaborative query programming problem-solving across different languages.

References

[1] Marjan Laal and Seyed Mohammad Ghodsi. Benefits of collaborative learning. *Procedia-social and behavioral sciences*, 31:486–490, 2012.

[2] Heisawn Jeong, Cindy E Hmelo-Silver, and Kihyun Jo. Ten years of computer-supported collaborative learning: A meta-analysis of cscl in stem education during 2005–2014. *Educational research review*, 28:100284, 2019.

[3] Geoffrey Herman, Yucheng Jiang, Yueqi Jiang, Seth Poulsen, Matthew West, and Mariana Silva. An analytic comparison of student-scheduled and instructor-scheduled collaborative learning in online contexts. In *2022 ASEE Annual Conference & Exposition*, 2022.

[4] Yu-Tzu Lin, Cheng-Chih Wu, and Chiung-Fang Chiu. The use of wiki in teaching programming: Effects upon achievement, attitudes, and collaborative programming behaviors. *International Journal of Distance Education Technologies (IJDET)*, 16(3):18–45, 2018.

[5] Travis Faas, Lynn Dombrowski, Alyson Young, and Andrew D Miller. Watch me code: Programming mentorship communities on twitch. tv. *Proceedings of the ACM on Human-Computer Interaction*, 2(CSCW): 1–18, 2018.

[6] Juan P Ucan, Omar S Gomez, and Raul A Aguilar. Assessment of software defect detection efficiency and cost through an intelligent collaborative virtual environment. *IEEE Latin America Transactions*, 14(7):3364–3369, 2016.

[7] Gerry Stahl. A decade of cscl. *International Journal of Computer-Supported Collaborative Learning*, 10: 337–344, 2015.

[8] Chaima Jemmali, Erica Kleinman, Sara Bunian, Mia Victoria Almeda, Elizabeth Rowe, and Magy Seif El-Nasr. Maads: Mixed-methods approach for the analysis of debugging sequences of beginner programmers. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*, pages 86–92, 2020.

[9] Bo Jiang, Wei Zhao, Nuan Zhang, and Feiyue Qiu. Programming trajectories analytics in block-based programming language learning. *Interactive Learning Environments*, 30(1):113–126, 2022.

[10] Christopher Watson and Frederick WB Li. Failure rates in introductory programming revisited. In *Proceedings of the 2014 conference on Innovation & technology in computer science education*, pages 39–44, 2014.

[11] Despina Tsompanoudi, Maya Satratzemi, and Stelios Xinogalos. Evaluating the effects of scripted distributed pair programming on student performance and participation. *IEEE Transactions on education*, 59(1):24–31, 2015.

[12] Matthew West, Geoffrey L Herman, and Craig Zilles. Prairielearn: Mastery-based online problem solving with adaptive scoring and recommendations driven by machine learning. In *2015 ASEE Annual Conference & Exposition*, pages 26–1238, 2015.

[13] Joni Lämsä, Raija Hämäläinen, Pekka Koskinen, Jouni Viiri, and Emilia Lampi. What do we do when we analyse the temporal aspects of computer-supported collaborative learning? a systematic literature review. *Educational Research Review*, 33:100387, 2021.

[14] Misty L Loughry, Matthew W Ohland, and David J Woehr. Assessing teamwork skills for assurance of learning using catme team tools. *Journal of Marketing Education*, 36(1):5–19, 2014.

[15] What is learning analytics?, Mar 2021. URL `https://www.solaresearch.org/about/what-is-learning-analytics/`.

[16] Leonardo Silva, António José Mendes, and Anabela Gomes. Computer-supported collaborative learning in programming education: A systematic literature review. In *2020 IEEE Global Engineering Education Conference (EDUCON)*, pages 1086–1095. IEEE, 2020.

[17] Michal Kompan and Maria Bielikova. Enhancing existing e-learning systems by single and group recommendations. *International Journal of Continuing Engineering Education and Life Long Learning*, 26(4): 386–404, 2016.

[18] Matthew Berland, Don Davis, and Carmen Petrick Smith. Amoeba: Designing for collaboration in computer science classrooms through live learning analytics. *International Journal of Computer-Supported Collaborative Learning*, 10:425–447, 2015.

[19] Owen HT Lu, Jeff CH Huang, Anna YQ Huang, and Stephen JH Yang. Applying learning analytics for improving students engagement and learning outcomes in an moocs enabled collaborative programming course. In *Learning Analytics*, pages 78–92. Routledge, 2018.

[20] Marielle Dado and Daniel Bodemer. A review of methodological applications of social network analysis in computer-supported collaborative learning. *Educational Research Review*, 22:159–180, 2017.

[21] Bodong Chen, Monica Resendes, Ching Sing Chai, and Huang-Yao Hong. Two tales of time: Uncovering the significance of sequential patterns among contribution types in knowledge-building discourse. In *Learning Analytics*, pages 20–33. Routledge, 2018.

[22] Kai-Tai Tang, Howard Leung, Taku Komura, and Hubert PH Shum. Finding repetitive patterns in 3d human motion captured data. In *Proceedings of the 2nd international conference on Ubiquitous information management and communication*, pages 396–403, 2008.

[23] K Sasirekha and P Baby. Agglomerative hierarchical clustering algorithm-a. *International Journal of Scientific and Research Publications*, 83(3):83, 2013.

[24] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[25] Nigel Bosch and Luc Paquette. What's next? sequence length and impossible loops in state transition measurement. *Journal of Educational Data Mining*, 13(1):1–23, 2021.