International Journal of Artificial Intelligence in Education
https://doi.org/10.1007/s40593-022-00302-5

ARTICLE

# Investigating the Relationship Between Dialogue States and Partner Satisfaction During Co-Creative Learning Tasks

**Amanda E. Griffith**[1] · **Gloria Ashiya Katuka**[1] · **Joseph B. Wiggins**[1] · **Kristy Elizabeth Boyer**[1] · **Jason Freeman**[2] · **Brian Magerko**[2] · **Tom McKlin**[3]

Accepted: 6 July 2022
© International Artificial Intelligence in Education Society 2022

## Abstract

Collaborative learning offers numerous benefits to learners, largely due to the dialogue that is unfolding between them. However, there is still much to learn about the structure of collaborative dialogue, and especially little is known about co-creative dialogues during learning. This paper reports on a study with learners engaged in co-creative tasks where the learners wrote code to create a song and while engaging in textual dialogue as they did so. After gathering the textual dialogue and the actions within the interface, we learned a hidden Markov model (HMM) to reveal co-creative states. The seven-state model revealed four states primarily composed of coding actions that included browsing the curriculum documents, working in the code editor, compiling the code successfully, and receiving a compile error. The remaining three states are primarily composed of dialogue that can be characterized as social, aesthetic, and technical dialogue. Next, we analyzed the relationships between the co-creative states revealed by the HMM and students' partner satisfaction scores from a post-survey. The results reveal the relative frequency of actions in certain states and some transitions between states were predictive of partner satisfaction. For example, partner satisfaction was negatively associated with the *Compilation Error* state and with the relative frequency of transitions from the *Curriculum Browsing* state to the *Code Editing* state. Partner satisfaction was also negatively associated with the relative frequency of transitions from the *Aesthetic Dialogue* state to the *Technical Dialogue* state and the *Code Editing* state. This line of investigation reveals how co-creative processes are associated with partner satisfaction, and holds the potential to inform scaffolding for collaborative learning.

**Keywords** Collaborative learning · Dialogue · Co-creativity · Hidden Markov model · Collaborative coding · Computational music remixing

✉ Amanda E. Griffith
amandagriffith@ufl.edu

Extended author information available on the last page of the article

Published online: 01 August 2022

## Introduction

Collaborative learning, in which two learners work together on a shared problem or goal, offers many benefits: it has been shown to increase learner interest in solving problems during online tutoring (Arroyo et al., 2017), decrease attrition rate in MOOCs (Ferschke et al., 2015), and improve critical thinking skills (Gokhale, 1995). The interactions that occur during collaborative learning can be observed in many ways, such as gaze synchronization (Schneider & Pea, 2014; Dich et al., 2018); body proximity and posture (Radu et al., 2020; Dich et al., 2018); and dialogue (Samoilescu et al., 2019; Carpenter et al., 2020; Snyder et al., 2020), which plays a fundamental role. Dialogue mediates joint problem solving and knowledge building by allowing collaborators to draw attention to problems and propose solutions (Swain, 2000). Research into collaborative dialogue has focused on detecting when students are off-task (Carpenter et al., 2020), analyzing the role of questions in collaborative computational modeling (Snyder et al., 2020), predicting problem-solving modes (Rodríguez & Boyer, 2015), and identifying issues during collaboration using dialogue features (Goodman et al., 2005).

While many forms of collaborative dialogue have been well studied, co-creative dialogues—collaborative dialogues that occur during co-creative activities—are understudied. Within co-creative activities, people, or people and machines, work together and share responsibility for creating an artifact (Glăveanu, 2011; Kantosalo et al., 2014). How key dialogue phenomena manifest within co-creative dialogues for learning is an open research question, and understanding these processes is an essential step toward to supporting collaborative co-creation in education.

To address this need, this paper investigates co-creative dialogues in the domain of computational music remixing. Specifically, we examined dialogue and system interactions between pairs of high school students learning to code through remixing musical samples. Studying the collaborative dialogues of pairs as they engage in computational music remixing provides insight into the unique exchanges that occur as they negotiate aesthetic decisions while navigating technical constraints. This work examines the following research questions in the context of computational music remixing with high school learners:

**RQ1**: What dialogue states characterize the co-creative process between learners, and how do learners transition between these dialogue states?
**RQ2**: What are the relationships between the outcome of partner satisfaction and the dialogue that comprised the co-creative process?

To investigate these questions, we first labeled the dialogue with dialogue acts and then modeled dialogue states with a hidden Markov model (HMM) over those sequences of dialogue acts interleaved with events in the computational music remixing interface. The HMM distinguished seven hidden states, three characterized by conversation and four characterized by task actions. After learning the seven-state HMM, we used the relative frequency of each pair's states

and their transitions to conduct a regression analysis to predict the partner satisfaction ratings from the post-survey. The results from this analysis reveal that the relative frequency of actions in certain states and some transitions between states were predictive of partner satisfaction. We found that partner satisfaction was negatively associated with the relative frequency of the *Compilation Error* state and with the relative frequency of transitions from *Curriculum Browsing* → *Code Editing*. Partner satisfaction was also negatively associated with the relative frequency of transitions from *Aesthetic Dialogue* → *Technical Dialogue* and *Aesthetic Dialogue* → *Code Editing*.

This work is the first to model the dialogue mechanisms of high school learners' co-creative interactions and provide insights into the relationship between these interactions and partner satisfaction. Based on these findings, we provide suggestions for intelligent systems to support co-creativity. We can move toward intelligent support of co-creative learning by modeling co-creative dialogue and its relationship with partner satisfaction.

## Related Work

Collaborative learning is a complex process involving both cognitive and social dimensions. Dillenbourg (1999) broadly defines collaborative learning as "a situation in which two or more people learn or attempt to learn something together". While there are a variety of more explicit definitions for collaborative learning, most tend to include some form of *creation of shared knowledge by groups while working together towards a common goal* (Davidson & Major, 2014; Roschelle & Teasley, 1995). The notion of working together towards a common goal is also part of collaborative problem solving. Graesser et al. (2018) provide nuanced descriptions of the features that distinguish collaborative problem solving from other forms of collaboration which include solving a novel problem, objective accountability of solution quality during problem solving, differentiation of roles, and interdependency among team members.

Understanding human-human collaboration is essential for developing AI systems that support collaborative learning. For many years researchers have attempted to characterize collaboration through identifying group processes and interaction patterns. Work from Bales and Strodtbeck (1951) provides an early example of investigating how group processes during problem-solving were distributed over time, finding that groups move through phases of collaboration and that different group processes characterize these phases. Since then, researchers have analyzed many types of collaborative interactions using a variety of coding schemes. Research on interaction processes generally uses sequences of low-level codes (e.g., dialogue acts) to identify patterns and higher-level constructs, such as cycles of turn taking (Roschelle, 1992), ideal communication cycles (Tschan, 1995), knowledge sharing events (Soller & Lesgold, 2007), inquiry threads (Zhang et al., 2007), and collaborative episodes (Chng et al., 2020). Identifying patterns in group processes allows researchers to distinguish differences between effective and ineffective groups. Tschan (1995) found that high-performing groups had a higher proportion of ideal

communication cycles (i.e., beginning with a discussion about task preparation and ending with an evaluation) than low-performing groups. Soller (2001) identified collaborative learning behaviors that were exhibited by effective collaborative learning teams. Chng et al. (2020) explored how individual and collaborative patterns were associated with individual outcomes such as frustration, technical proficiency, and time spent in the makerspace.

While focus on collaborative learning has been increasing, not enough is understood about it in K-12 contexts. Specifically, while practices such as collaborative computing have been identified as core practices in the US's K-12 CS Framework (K–12 Computer Science Framework, 2016), most research into collaborative programming happens in undergraduate computer science courses. Researchers have found that undergraduate learners who participate in collaborative programming produce higher quality code, greater confidence in their work, and are more likely to pass their programming courses (Braught et al., 2011; McDowell et al., 2002). Research into K-12 collaborative programming practices include comparing the benefits of pair programming for middle school students to working individually (Denner et al., 2014), exploring the different ways pairs' dialogue, suggestions, and roles affect collaboration (Tsan et al., 2018; Tsan et al., 2018), asking how attitude toward collaboration and prior knowledge of programming influences interactions (Campe et al., 2020), and investigating how pair programming improves confidence and performance in primary school girls (Zhong et al., 2016).

Another practice identified in the US's K-12 CS Framework is creating computational artifacts, which is a process that "embraces both creative expression and the exploration of ideas to create prototypes and solve computational problems. Students create artifacts that are personally relevant or beneficial to their community and beyond". (K–12 Computer Science Framework, 2016) Studies have consistently found benefits to incorporating more creative activities into computer science classes, such as increased motivation, engagement, and content knowledge.

Knobelsdorf and Romeike (2008) identified two types of students interested in computer science and programming. The first type viewed computer science as "fun, creative, and autonomous", and the second type was more interested in acquiring knowledge and solving problems. The first group found their computer science classes disappointing, suggesting that more opportunities for creativity in computer science courses may be necessary to prevent loss of interest. While studying the effects of computational music remixing on attitudes and knowledge of computer science on high school students, Freeman et al. (2014) found students value these creative and personally relevant aspects, and their results suggested that these aspects may be even more engaging for female and minority students. These findings were supported by Magerko et al. (2016), who observed that while all groups of students showed significant increases in intent to persist in computing and content knowledge, the increase in female student interest in computing was significantly higher than the increase in male interest.

Despite the benefits of collaboration and creativity in computer science education, there is little research into the processes of co-creativity in computer science classrooms. While studying co-creative processes and how they impact concept formation, Matsumae et al. (2021) found that, compared to non-interactive and

interactive non-co-creative processes, the co-creative process enhances concept formation depth but also requires more time and a greater cognitive load. This finding indicates that we cannot assume the same processes are involved between collaborative programming and co-creative programming. In fact, Bales and Strodtbeck (1951), while studying phases of collaboration among different activities, notes how the phase order in "certain types of creative problems" was the reverse of what was predicted, and that these tasks may require a "different sort of phase hypothesis". When designing systems to support co-creative learning, it is essential to understand the co-creative processes that occur and how they might impact learner outcomes.

There is a growing interest in investigating partner satisfaction rather than focusing exclusively on individual outcomes in co-creative domains. Historically, partner satisfaction has been seen as important in reflecting the extent to which collaborators are engaged or adding value to the collaboration (Waddock & Bannister, 1991). Recent work supports this finding: Campe et al. (2020) found that students' attitudes about each other influenced the extent to which they contributed during pair programming. Similarly, Katuka et al. (2021) found that dialogue acts reflecting higher participation were associated with higher partner satisfaction during pair programming activities. Furthermore, there seems to be a relationship between partner satisfaction and how student partners talk with one another (Katuka et al., 2022). However, little is know about the relationship between partner satisfaction and higher-level processes in co-creative group interactions.

Hidden Markov models (HMM) are well-suited to learning higher-level constructs from observable interactions during collaboration in computer science. Soller and Lesgold (2007) compared five computational approaches—finite state machines, rule learners, decision trees, plan recognition, and hidden Markov models—for modeling knowledge sharing during collaborative learning. They found that the HMM performed significantly better than the other approaches. In the domain of computer science learning, Boyer et al. (2009) learned an HMM from a corpus of human-human introductory computer science tutoring dialogues. They demonstrated that an HMM could be used to identify tutorial dialogue strategies from the structure of the dialogue act sequences. Later, Boyer et al. (2011) used both dialogue acts and task actions from human-human tutoring and demonstrated that HMMs could identify meaningful high-level structures (i.e., hidden states) that were correlated with learning outcomes when the low level codes (i.e., observation symbols) that made up these states were not. In addition to computer science tutoring, this method has also been applied to collaborative programming processes. Rodríguez and Boyer (2015) used HMMs to compare the problem-solving approaches of individual students and pairs of students during programming activities. The HMM was able to highlight the similarities and differences of the types of states each moved through and where the pairs employed different problem-solving patterns and strategies than the individuals. For instance, only pairs had the *program planning* and *program building* states. Researchers noted that the *program planning* state might be most representative of the collaborative aspect of the task. Additionally, they noted that pairs were more likely to persist in the same problem-solving state for longer than individuals.

In summary, the prior research has established that in CS education, collaboration is beneficial, creative coding can help motivate and engage students, and HMMs can be used to identify the higher-level processes of group interactions. The work presented in this paper builds on previous research to identify the group processes of collaboration and creativity in learning. While earlier research has focused on exploring these processes separately or in other learning domains, we focus on learning the processes of co-creative learning in K-12 computer science education using an HMM. Additionally, while most research into collaborative learning focuses on learning outcomes or performance on an activity or artifact, we explore the relationship between co-creative processes and partner satisfaction.

## Methods

This work analyzes a corpus collected from a study conducted in high school computer science classes. The corpus consists of textual student-student dialogue gathered while students worked together in pairs within the EarSketch learning environment, an online interface for developing computational music (Fig. 1). In prior studies, students, especially those in currently underrepresented groups in computing, who used EarSketch had significant positive results related to content knowledge and attitudes towards computing, (Magerko et al., 2016). The EarSketch interface includes a code editor and a digital audio workstation that allows users to access the music they have written (Freeman et al., 2015). The EarSketch interface also

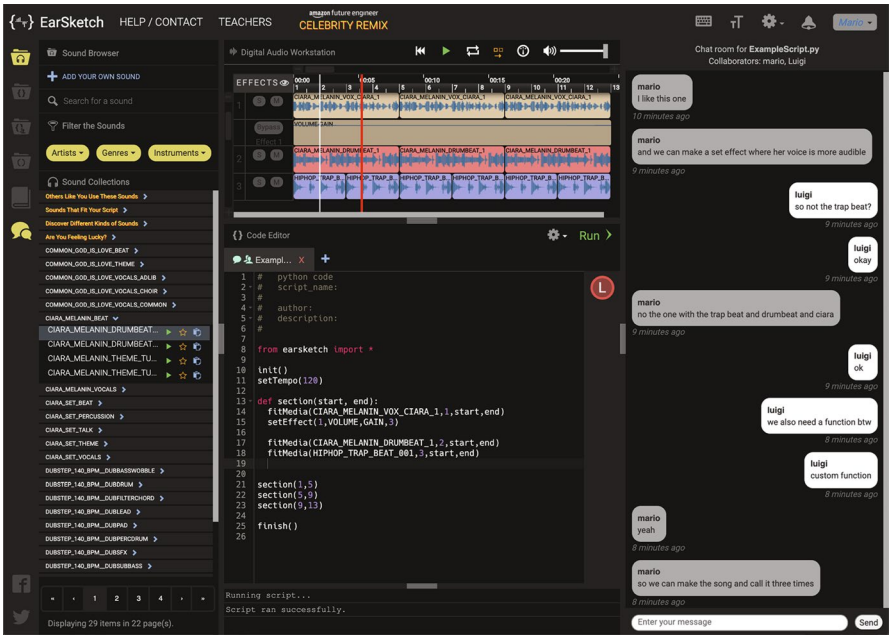


**Fig. 1** The modified EarSketch environment with chat window used during the study

features a content manager with samples, which students can use to create music. A curriculum tab provides helpful resources associated with the class. Both students in each pair had access to all tools, allowing both to contribute to the code simultaneously. In this study, the interface included a chat box for partners to communicate with one another. We logged all pairs' textual dialogue, the activity in the code editor, the items viewed in the curriculum tab, and the outcome of the students running their scripts (e.g., successful compile or error received).

There were 140 participants from eight public high schools in two districts in the southern United States. More than half of the schools had a student population of majority (> 50%) White students; one school was majority Black; two schools had a substantial (> 25%-35%) Latinx population; one school had a substantial Asian population. Students were in grades 10-12, which is around 15-18 years old. To encourage the students to communicate through the textual chat interface (Fig. 1), teachers placed the pairs either in separate rooms or different areas of the same room. Students collaborated synchronously for an average of 48 minutes to remix musical samples and create an original song or ringtone. Nine pairs split their work across two class days. We only included the first day's dialogue because including the dialogue from both days would unevenly weight the patterns of those nine pairs while training the hidden Markov models, but concatenating the dialogues from the two separate days would change the natural beginning, middle, and end of the sequences.

After their collaboration, students were given a post-survey about their satisfaction with their partner, which we will discuss in a later section. Some students were also given a pre-survey that asked students to describe their experience, confidence, and enjoyment in both computing and music. However, because pre-surveys were not administered during the first sets of classroom studies, we only have pre-survey data for 44 of the 136 students. Therefore, we did not use the pre-survey in our analyses.

### Dialogue Taxonomy

We extracted the students' textual dialogue into a corpus made up of 4,533 unique, raw utterances. We excluded sessions that included groups of three (in the case of an odd number of students) and removed two sessions that contained exclusively off-task, joking, offensive, or gibberish content. The remaining textual dialogue corpus includes 68 sessions (136 students) and 3305 utterances, with a mean of 48 utterances per session ($SD = 35$, $Min = 4$, and $Max = 214$).

To analyze the utterances within these co-creative dialogues, we developed and applied a dialogue act taxonomy that included 16 dialogue act labels to effectively capture the collaborative/technical and creative/aesthetic dialogue processes within a co-creative domain. Our dialogue act taxonomy consisted of relevant dialogue acts identified from existing taxonomies of collaborative coding (Rodríguez et al., 2017a) and improvisational theatre (Fuller & Magerko, 2010; Fuller & Magerko, 2011). From Rodríguez et al. (2017a), we included the labels Statement, Acknowledgment, Positive Feedback and Non-positive Feedback. From Fuller and Magerko (2010) and Fuller and Magerko (2011), we included Proposal (introducing new ideas), Directive

(directing a collaborator), Proposal Acceptance or Rejection, and Directive Acceptance or Rejection. Additional dialogue acts labels were generated to capture the remaining dialogue acts within the corpus, including Social, Passing Responsibility, Confusion, Seeking Feedback, Closing, and Code/Link. Table 1 shows these dialogue act labels. Our resulting dialogue act taxonomy captured the technical and aesthetic dimensions within the co-creative dialogue.

Three independent annotators, all graduate students, began the tagging process and labeled an initial subset of the corpus (approximately 190 utterances). After refining the dialogue act labels and descriptions, two annotators tagged another subset of the dialogue corpus (approximately 185 utterances) to arrive at a sufficiently high inter-rater reliability (kappa $\geq 0.70$). Annotator A then labeled the remaining 2851 utterances, and annotator B labeled 918 utterances, resulting in a kappa of 0.76, indicating substantial agreement for overlapping utterances (Landis & Koch, 1977).

### Hidden Markov Model

We implemented a hidden Markov model (HMM) to analyze the learners' interactions and model the co-creative sequences (Rabiner & Juang, 1986) after compiling the lists of sequential observation symbols that represent the collaborative interactions. We chose to use an HMM because we were interested in the hidden discourse states of co-creative dialogue. In an HMM, sequences of *observation symbols* represent observable events such as textual messages and coding actions. In Fig. 2 the circles with the dialogue act labels represent the observation symbols. Hidden states are influences upon those observation sequences. Each hidden state is characterized by an *emission distribution*, which is a probability distribution over observation symbols. When the model is learned, every observation is modeled as having been "generated" by a hidden state. In Fig. 2, the hidden states are represented by the unlabeled squares, and the arrow from the hidden states to the observations indicates the relationship of the observation having been generated by the hidden state. Each hidden state also has a set of transition probabilities and an initial state probability. The transition probabilities indicate how likely the model is to either continue in that state or transition to another state are represented by the arrows from one hidden state to the next in Fig. 2. The initial state probability that indicates the likelihood of the Markov chain beginning in that state. The HMM also has a stationary distribution that indicates the proportion of interactions that occur in each state over the long run. By training an HMM on the data from this study, we can take a high-level view of the learners' interactions and uncover generalizable patterns in the co-creative process.

The labeled dialogue and task actions are the observation symbols in this model. There are 20 distinct possible observation symbols—16 dialogue acts (Table 1) and the following actions:

- **curriculum:** The student accessed the curriculum or moved between lessons.
- **edit:** Any consecutive insertion or removal of characters in the code editor.
- **success:** Each time the script was run successfully.

**Table 1** Taxonomy of co-creative dialogue acts

| Dialogue act label | Rel. Freq. | Description | Example |
|---|---|---|---|
| Statement | 17.1% | Utterance of information or explanation, or something that exists in the coding workspace | *well we also have to make a loop* |
| Social | 14.1% | A general salutation, off-task comment, or display of remorse. Plays some social function not captured in the other tags | *how are you?* |
| Proposal | 12.3% | An assertion of creativity, related to code or music, for the partner to consider. | *we should do some beats in the background* |
| Directive | 11.5% | An utterance used to set task responsibilities for one or both partners | *We should focus on the custom function first* |
| Confusion | 10.4% | Seeking help, or expressing confusion, lack of knowledge, or uncertainty | *What are those variables for?* |
| Acknowledgement | 6.35% | Acknowledging the content of the previous utterance or series of utterances | *yeah* |
| Passing responsibility | 6.17% | Passing creative or technical choice to partner | *Do you know what sounds you would like to use?* |
| Proposal acceptance | 5.67% | Accepting a partner's addition or assertion to the co-creative mental model shared by both partners | *yeah jazzand dubstep sounds fine* |
| Positive feedback | 5.29% | Positive response relating to something the partner accomplished within the scope of the task | *I liked the piano thing you did* |
| Directive acceptance | 3.97% | Response to a partner accepting the dictation of flow or direction of project | *ok i will figure out a makebeat* |
| Seeking feedback | 2.44% | Calling attention to or requesting comment from partner regarding one's creative contribution or state of project | *Check this out* |
| Non-positive feedback | 2.29% | Non-positive response relating to something incorrectly done by the partner within the scope of the task | *it doesnt sound as good as i thought it would* |
| Closing | 1.00% | Partner asserts or offers to complete the program and finish the session | *there we are done* |
| Directive rejection | 0.53% | (Often a response to Directive) Response to a partner rejecting the dictation of flow or direction of project | *no its fine* |
| Proposal rejection | 0.53% | (Often a response to Proposal) Rejection of a partner's addition or assertion to the co-creative mental model shared by both partners | *I don't think so* |
| Code/Link | 0.21% | Code statements or snippets (Music sample titles are not necessarily code), URL or other hyperlink to material or resources (curriculum entries, links to Stack Overflow) | *Applied API function make- Beat()* |

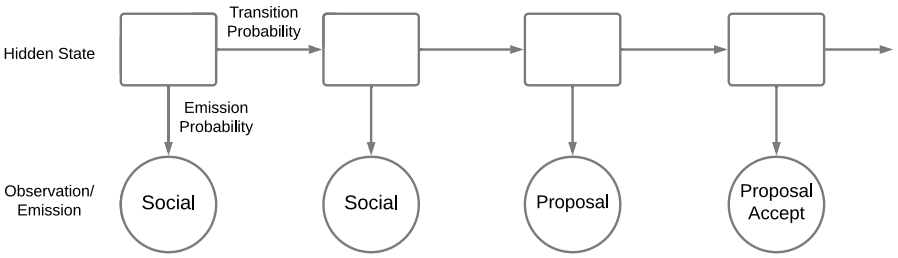


**Fig. 2** Example HMM diagram

- **error:** Each time an error was received when the script was run.

We represented each of the 68 collaborative dialogues as a sequence of these observation symbols and trained an HMM on these sequences. We did not model time between actions, nor did we model which of the two students performed each action.

## Partner Satisfaction Survey

Each participant filled out a post-survey where they rated their partner on seven items using a 4 point scale of Strongly Disagree (1) to Strongly Agree (4) with no neutral answer. Figure 3 shows the summary of these results. We conducted a Principal Component Analysis (PCA) on the survey item responses that we collected. The PCA results suggest that these seven items should be considered one PCA cluster, so we combined these seven items to obtain a single *partner satisfaction* for each student. We then summed the two *partner satisfaction* scores of the students in each pair to form a *combined satisfaction* score. The *combined satisfaction* score provided a group metric that indicates overall partner satisfaction. The range

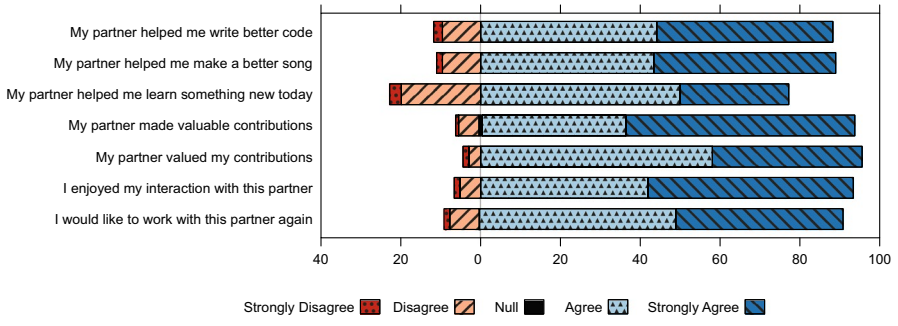


**Fig. 3** Post-survey responses on perceived partner contribution

for possible *combined satisfaction* scores is from 14 to 56, with actual scores that ranged from 19 to 56 ($M = 46.4$, $SD = 6.17$).

The post-survey also asked students to assess their partner's coding knowledge in response to the prompt *My partner's coding knowledge was:* with a scale of *Much less than mine*, *Somewhat less than mine*, *About the same as mine*, *Somewhat better than mine*, and *Much better than mine*. The post-survey also included two open response questions. The first was a follow-up to the *My partner made valuable contributions* item, *—If so, what was the most valuable contribution your partner made?* The second open response question was a follow-up to the *I would like to work with this partner again* item, *—How come?*

## Hidden Markov Model of Co-creative Dialogue

### Results

We used a hidden Markov model to answer *RQ1: What are the dialogue states that characterize the co-creative process between learners, and how do learners transition between these dialogue states?* One of the major decisions when learning an HMM is choosing how many hidden states to include. Sometimes the number of hidden states is known or strongly implied by physical or theoretical constraints of an application, but in our case, we had no prior notion of how many co-creative dialogue states would emerge. To identify the best number of hidden states $n$, we used leave-one-out cross-validation and compared the average Akaike information criterion (AIC) score for each number of hidden states (Akaike, 1974). We compared models using $n = 4$ to $n = 9$ states, finding the best AIC scores with $n = 6$ and $n = 7$. We then trained ten models each for both six and seven states and picked the single best model for each $n$ using log-likelihood. The best models for each $n$ were nearly identical: one of the dialogue states for the six-state model split into two dialogue states in the seven-state model, with the remaining states being the same. We opted to move forward with the 7-state HMM.

The HMM analysis suggests seven hidden states within the co-creative dialogues (see Fig. 4). These hidden states are characterized by emission probability distributions over the observation symbols and must be interpreted as a whole. We interpret the learned hidden states as follows:

- **Social Dialogue:** In this state, *social* dialogue acts are emitted with a high probability (79%). Due mainly to greetings and other social dialogue at the outset of collaboration, around 90% of sessions begin in this state.
- **Aesthetic Dialogue:** This state is characterized by the observation symbols *proposal* (34%) and *proposal acceptance* (17%), which are the dialogue acts involving assertions and acceptances of creativity. The dialogue that belongs to this state usually involves discussing some aspect of the music.
- **Curriculum Browsing:** In this state, the observation symbol *curriculum*, which involves accessing the curriculum documents, is emitted with a very

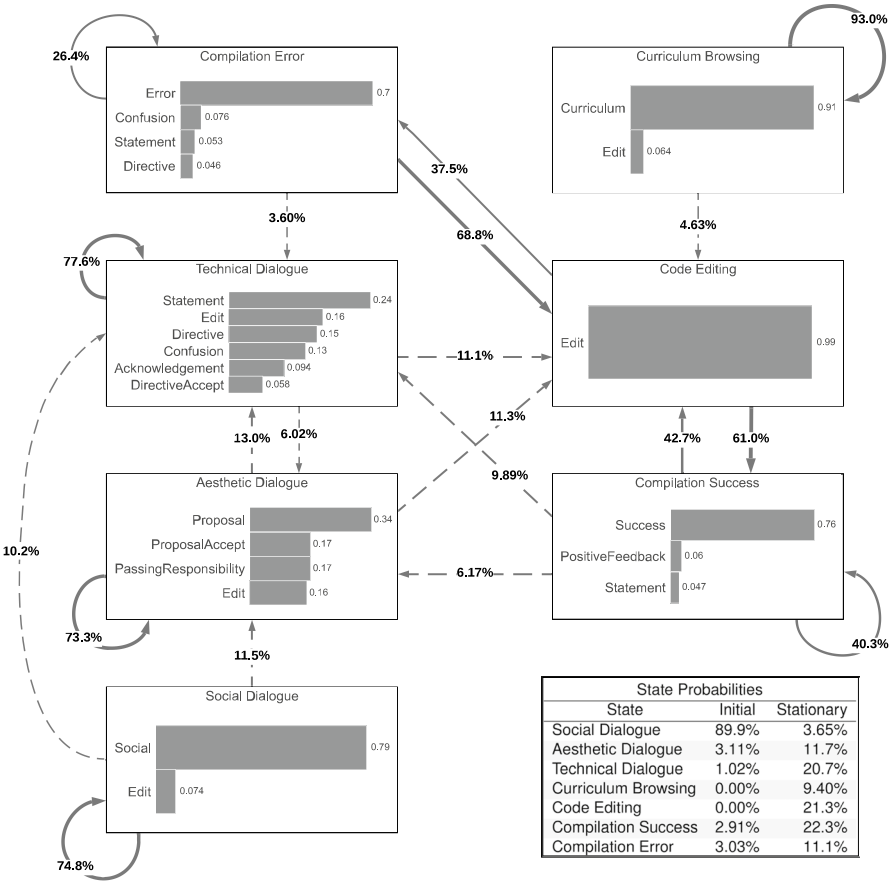


**Fig. 4** The learned HMM with $n = 7$ hidden states. Rectangles represent hidden states, and probability distributions within each rectangle indicate emission probabilities. Arrows among hidden states represent transition probabilities. The lower right table displays initial probabilities (probability of a dialogue session beginning in that state) and stationary distribution (proportion of interactions that occur in each state over the long run)

high probability (91%). A student browsing through several curriculum documents rather than accessing specific relevant documents characterizes this state.

- **Code Editing:** The observation symbols from this state are almost entirely code editing, which is emitted with a 99% probability.
- **Technical Dialogue** The observation symbols that characterized this state involved dialogue acts of *statement* (25%), *directive* (15%), *confusion* (13%), *acknowledgement* (9%), and *directive acceptance* (6%). The dialogue that belongs to this state usually involves discussion of code features or task requirements.

- **Compilation Success:** Mostly characterized by students running the code successfully, this state involves some *positive feedback* (6%) and *statement* (5%) dialogue acts.
- **Compilation Error:** Mostly characterized by students receiving an error when running the code, this state also involves some *confusion* (8%), *statement*(5%), and *directive*(5%) dialogue acts.

This model revealed three distinct states of conversation in these co-creative interactions: *Social Dialogue*, *Aesthetic Dialogue*, and *Technical Dialogue*. The *Social Dialogue* state usually occurs at the beginning of the interaction, but can occur throughout and usually includes some rapport building and off-task discussions. Utterances in the *Aesthetic Dialogue* state usually involved discussing different aspects of the music such as instruments, tempo, genre, and even what artist to emulate. Utterances in the *Technical Dialogue* state were typically about task requirements and code. This model also revealed four hidden states focused on coding: *Curriculum Browsing*, *Code Editing*, *Compilation Success*, and *Compilation Error*. The sessions never begin in the *Curriculum Browsing* state, and no other states consistently lead to it. Every state, excluding the *Social Dialogue* state, has a significant chance to lead to the *Code Editing* state, and 21.3% of the actions occur in the *Code Editing* state. *Code Editing* transitions to a *Compilation Success* or a *Compilation Error* state with a probability of 98.5%. The *Compilation Success* state is likely to transition to *Code Editing* (42.7%), *Technical Dialogue* (9.89%), and *Aesthetic Dialogue* (6.17%). The *Compilation Error* state is likely to transition to is *Code Editing* (68.8%), and then *Technical Dialogue* (3.60%).

## Discussion of Hidden States

### Dialogue States

The HMM revealed how co-creative pairs moved among collaborative dialogue states characterized by conversation and task actions. Of the seven hidden states identified by the HMM, three were composed primarily of dialogue acts. *Social Dialogue* was the most likely state for pairs to start in, primarily composed of greetings and off-task dialogue. This is a typical feature of collaborative dialogue, prefacing discussion with periods of rapport building in which the partners become more familiar with each other (Ogan et al., 2012). After leaving this initial *Social Dialogue* state, we found that the conversation was nearly twice as likely to move directly to the *Aesthetic Dialogue* state (60%) as the *Technical Dialogue* state (31%). In the *Aesthetic Dialogue* state, pairs brainstorm and exchange dialogue related to the musical aspects of their co-creative task, such as genre and types of sounds. The *Technical Dialogue* is where pairs begin planning how to accomplish their aesthetic goals.

In the excerpt in Table 2, the pair sets their goal of creating a "dubstep" song and then debates how they would accomplish that in their code. This transition from *Aesthetic Dialogue* → *Technical Dialogue* accounted for 13.0% of the transitions across the sessions. In contrast, the transition from *Technical Dialogue* → *Aesthetic*

**Table 2** Excerpt from a student collaborative dialogue showing a pair's conversation transitioning from *Aesthetic Dialogue* → *Technical Dialogue State*. Each dialogue state was determined automatically using the HMM presented in this work

| State | Action | User | Text |
|---|---|---|---|
| Aesthetic | Passing responsibility | **Student 1** | what you want to do |
| Aesthetic | Proposal | **Student 2** | lets do dubstep cause its fire |
| Aesthetic | Proposal accept | **Student 1** | i feel you |
| Technical | Confusion | **Student 1** | what did she say how many variables |
| Technical | Statement | **Student 2** | 3 |
| Technical | Directive | **Student 1** | ok lets do this |
| Technical | Confusion | **Student 2** | ngl im kinda lost already so im sorry |
| Technical | Confusion | **Student 1** | i dont know what to do |
| Technical | Confusion | **Student 2** | me either ima ask for help |

*Dialogue* was less than half that at 6.02%. This observation suggests that most pairs tend to decide on what they want to make before they move on to making it.

We used a ridgeline plot (Fig. 5) to visualize where the hidden states occur in the sessions. The highest points are where the interactions in each state occur most frequently but are not relative to the other states' frequencies. The pattern we observed using all 68 sessions was that the highest frequency of *Social Dialogue* occurred at the beginning, followed by the highest frequency of *Aesthetic Dialogue*. The highest frequency of *Technical Dialogue* occurred just before the editing and debugging process began, but *Technical Dialogue* was more consistent throughout the sessions than *Social Dialogue* or *Aesthetic Dialogue*. Overall, dialogue appears to be more frequent at the beginning of the sessions and decreases as the editing and debugging processes take precedence.

### The Debugging Process and the Conversation it Inspires

The remaining four states are primarily composed of actions in the interface:

- Browsing curriculum documents (*Curriculum Browsing* state)
- Working in the code editor (*Code Editing* state)
- Encountering compilation errors (*Compilation Error* state
- Successfully compiling code (*Compilation Success* state)

The transitions between the *Code Editing*, *Compilation Success*, and *Compilation Error* states demonstrate the movement between collaborative states that occur during debugging, and they offer insights into how co-creative conversations unfold around debugging. The *Code Editing* state primarily transitioned to the *Compilation* states (*Error* or *Success*), with only four transitions to the *Social Dialogue* state in the entire dataset and zero transitions to the *Aesthetic Dialogue* or *Technical Dialogue* states. After the *Compilation Success* state, students were most likely to go back to the *Code Editing* (43%) state, but they sometimes

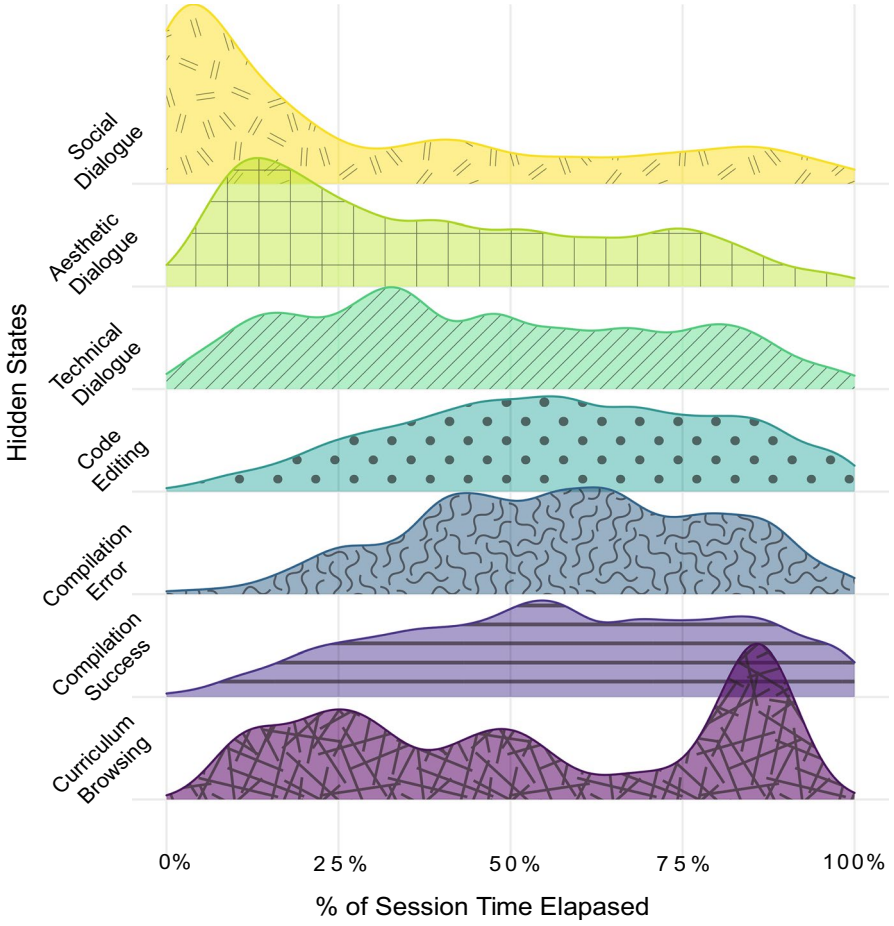


**Fig. 5** Hidden states as they occur throughout all 68 sessions. The widest part of each curve corresponds to the most frequent occurrence of that state during the sessions. This frequency is not relative to other states' frequencies

transitioned back into the *Technical Dialogue* (10%) or *Aesthetic Dialogue* (6%) states. The excerpt in Table 3, depicts a pair successfully compiling their code and transitioning to the *Technical Dialogue* state, discussing the measure lengths. The pair then transitions to the *Aesthetic Dialogue* state, making proposals about what to do with sounds and overall song length. Later in the session, after another successful code compilation, the pair transitions straight to *Aesthetic Dialogue*, discussing other sounds to add to their artifact.

The *Compilation Success* state seems to be an inflection point in the co-creative process, in which the pairs may start to renegotiate some of the creative aspects of their code. In contrast, the *Compilation Error* state only transitions to *Code Editing* or *Technical Dialogue* states, suggesting partners who encounter

**Table 3** Excerpt from a student collaborative dialogue showing a pair's successful code compilation leading to Technical and Aesthetic Dialogue. Each dialogue state was determined automatically using the HMM presented in this work

| State | Action | User | Text |
|---|---|---|---|
| Success | Compile | **Student 1** | Success |
| Technical | Acknowledgement | **Student 2** | oh |
| Technical | Statement | **Student 1** | ohh its because they are 2 second each |
| Technical | Statement | **Student 2** | each measure isnt exactly 2 seconds |
| Technical | Statement | **Student 2** | its a little longer i think |
| Technical | Statement | **Student 1** | but when another is added to the 30 it becomes exactly 2 seconds longer |
| Technical | Statement | **Student 2** | measure 15 is at 28.5 seconds |
| Technical | Compile | **Student 2** | Success |
| Aesthetic | Proposal | **Student 1** | we can use a combination of sounds |
| Aesthetic | Proposal | **Student 2** | we should just leave it at 31 |
| Aesthetic | Proposal | **Student 2** | i think that will be fine |
| Aesthetic | Proposal accept | **Student 1** | yeah |
| | ... | ... | ... |
| Success | Compile | **Student 2** | Success |
| Aesthetic | Editing code | **Student 1** | ... |
| Aesthetic | Proposal | **Student 2** | We should put like a synth or something to that effect |
| Aesthetic | Proposal | **Student 2** | Add some like futuristic noises or airhorns or something |

errors focus on resolving their problems rather than discussing the aesthetic elements of the artifact.

## Model Relationship with Combined Satisfaction

### Using Hidden States and Transitions to Predict Combined Satisfaction

After learning an HMM, our next step was to identify how the hidden states of the HMM (their relative frequencies out of the collaborative sessions, and the transitions between the hidden states during those collaborative sessions) were associated with the outcome of *combined satisfaction*. This research explores *RQ2: What are the relationships between the outcome of partner satisfaction and the dialogue that comprised the co-creative process?* Our goal is to build a predictive model using these state and transition features as predictors. To identify which of the hidden states and their transitions predicted *combined satisfaction*, first we calculated the relative frequencies for each state and for the transitions among states.

To illustrate how we computed relative frequency, we take the example in Fig. 6, which depicts an HMM that includes six observations, *Social, Social, Passing Responsibility, Proposal, Proposal Accept, and Statement*, and their hidden states, *Social Dialogue, Social Dialogue, Aesthetic Dialogue, Aesthetic*

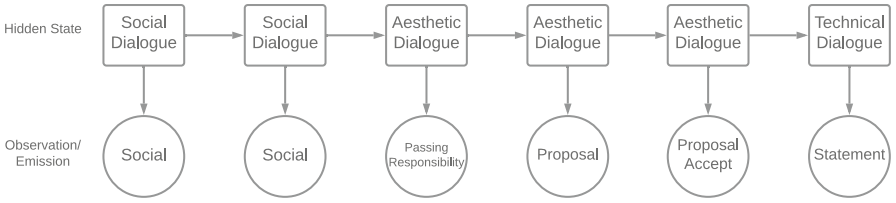


**Fig. 6** HMM transition example

*Dialogue, Aesthetic Dialogue, and Technical Dialogue*. If these six collaborative moves represented a pair's entire session, the session would have six observations, six (not necessarily distinct) hidden states, and five transitions. To compute the relative frequency for the *Social Dialogue* state, we count the number of times it occurred and divide by the total number of observations. In our simple example the model fit the *Social Dialogue* onto two of the six observation symbols, so the relative frequency of that state is $\frac{2}{6}$ or $\frac{1}{3}$. We computed this relative frequency for all states, so for our simple example $\frac{1}{2}$ of the interactions occurred in the *Aesthetic Dialogue* state, and $\frac{1}{6}$ of the interactions occurred in the *Technical Dialogue* state. Every other state in this example would have a 0% for the relative frequency of interactions that occurred in that state. This relative frequency is not equivalent to the elapsed time that was spent in each state as the HMM only considers the sequence order of the discrete observations and not the time duration.

We use the relative frequencies of the transitions to investigate how the flow of the interaction is associated with *combined satisfaction*. To calculate the relative frequency of each transition, we divide the number of transitions from each state to the next state (e.g., the transition from *Social Dialogue*→ *Social Dialogue* or the transition from *Aesthetic Dialogue*→ *Technical Dialogue*) by the total number of transitions in that collaborative session. In our simple example, the transition from *Social Dialogue*→ *Aesthetic Dialogue* occurs once out of the five transitions, so the relative frequency is $\frac{1}{5}$. Continuing this simple example, $\frac{1}{5}$ of the pair's transitions were from *Social Dialogue*→ *Social Dialogue*, $\frac{2}{5}$ were from *Aesthetic Dialogue* → *Aesthetic Dialogue*, and $\frac{1}{5}$ were from *Aesthetic Dialogue* → *Technical Dialogue*. Any other possible transitions would have a 0% transition relative frequency.

The model assumes that the predictors are not highly correlated, but we discovered that there was a strong correlation between the relative frequency of interactions in a specific hidden state and the relative frequency of self-transitions of that hidden state. Therefore, prior to model building we removed self transitions for each state. For example, the relative frequency of interactions in the *Social Dialogue* state and the relative frequency of self-transitions from the *Social Dialogue* → *Social Dialogue* state had strong positive linear relationship ($r(68) = .97$, $p < .0001$), thus we removed the self-transition from *Social Dialogue*→ *Social Dialogue* from the list of predictors. There were no self-transitions from the *Code Editing* state, so we chose to use the relative frequency of the hidden states instead of the hidden state self-transitions.

After computing the relative frequencies of states and state transitions for each pair's session, we treated those values as features within a regression model. We had 20 independent variables (predictors), which included the relative frequencies of the 7 hidden states and the relative frequencies of 13 transitions between the hidden states for each pair. The dependent variable was the *combined satisfaction* score for the pair. We used JMP (SAS Institute Inc. 2021), SAS's statistical analysis software suite, to conduct the regression analysis using best subset selection method to explore which of the relative frequencies of the hidden states and their transitions predicted the *combined satisfaction*. The best subset selection method identifies the best subset of significant predictors. It considers all possible models from the null model, which includes only the intercept and no predictive features, to $k$ predictors—$k$ being the number of independent variables—and chooses the best model for each size and then the best overall model from those models (SAS Institute Inc, 2020). We interpret the best model, which we selected based on the Bayesian Information Criterion (BIC) (Neath & Cavanaugh, 2012).

The significant predictors of *combined satisfaction*, along with their parameters estimated, are shown in Table 4. For ease of interpretation, we used centered and scaled predictors to provide standardized estimates so that we may compare the coefficients when we interpret the models. The results show that the transitions from *Curriculum* → *Code Editing*, *Aesthetic Dialogue* →*Technical Dialogue*, and *Aesthetic Dialogue* → *Code Editing* have a negative relationship with *combined satisfaction*. This means that the larger the relative frequency of these transitions that pairs have, the lower the *combined satisfaction*. Additionally, the greater the relative frequency of the interactions that occur in the *Compilation Error* state, the lower the *combined satisfaction*.

## HMM's Relationship with Combined Satisfaction

### Compilation Error State's Negative Relationship with Combined Satisfaction

The *Compilation Error* state had a negative relationship with *combined satisfaction*. The higher the relative frequency of interactions in the error state, the lower the *combined satisfaction* for the pair. Pairs that had more of their interactions in the *Compilation Error* state likely also had more issues completing their task, thus spending more time debugging. Prior research has established that, when compared to other programming tasks, students find the debugging process to be

**Table 4** Results of generalized regression using combined satisfaction as response ($n = 68$)

| Variable | Coefficient | Std error | $p$-value |
|---|---|---|---|
| Curriculum to code editing transition | -14.2 | 4.74 | 0.0026 |
| Aesthetic dialogue to technical dialogue transition | -20.5 | 8.81 | 0.0198 |
| Aesthetic dialogue to code editing transition | -15.2 | 6.88 | 0.0264 |
| Relative frequency of interactions in compilation error | -21.4 | 8.93 | 0.0164 |

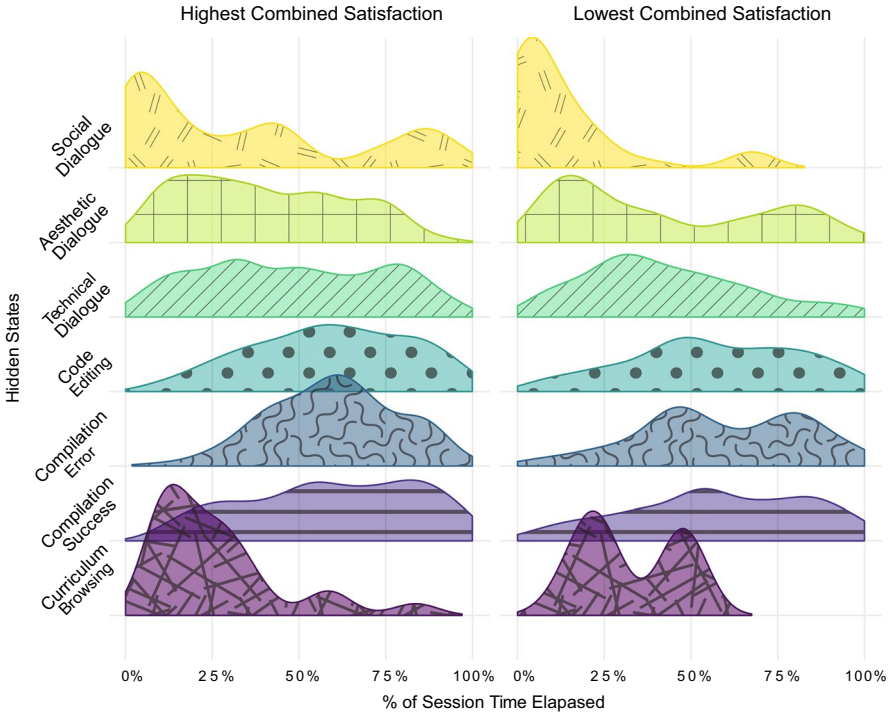


**Fig. 7** Hidden states as they occur throughout sessions for the 21 pairs with the highest *combined satisfaction* and 20 pairs with the lowest *combined satisfaction*. The widest part of each curve corresponds to the most frequent occurrence of that state during the sessions. This frequency is not relative to other states' frequencies

more tiring and less enjoyable when working in pairs (Chaparro et al., 2005). We compared the top and bottom one-third of pairs based on their *combined satisfaction* to identify their distinguishing features. There were 21 pairs, around 31%, with *combined satisfaction* greater-than-or-equal-to 50, and 20 pairs, around 29%, with *combined satisfaction* less-than-or-equal-to 44. We used a ridgeline plot (Fig. 7) to compare the pairs with the highest and lowest *combined satisfaction*. We noted that the pairs with the highest *combined satisfaction* had a single peak of *Compilation Error* state frequency around 60% through the session with most errors being resolved by the end of their sessions. In contrast, the pairs with the lowest *combined satisfaction* had two peaks of *Compilation Error* state frequency and more unresolved errors at the end of their sessions. An example of a lower rated pair that ends their session with compile errors and expressions of confusion can be seen in the excerpt in Table 5. Alternatively, in the excerpt in Table 6 we can see a pair with higher *combined satisfaction* ending the session with working code and positive feedback on their song.

One of the main differences between the interactions of the pairs with the highest and lowest *combined satisfaction* is that the pairs with higher *combined*

**Table 5** Excerpt from a student collaborative dialogue that ends their session without resolving errors. Each dialogue state was determined automatically using the HMM presented in this work

| State | Action | Student | Text |
|---|---|---|---|
| Error | Compile | **Student 1** | Unknown Identifier |
| Error | Passing responsibility | **Student 1** | do you want to change anything to the song ? |
| Error | Passing responsibility | **Student 2** | i'm not sure i have to listen to it so i can check |
| Error | Compile | **Student 2** | Unknown Identifier |
| Edit | Editing code | **Student 2** | .... |
| Error | Compile | **Student 2** | Unknown Identifier |
| Error | Compile | **Student 2** | Unknown Identifier |
| Edit | Editing code | **Student 2** | .... |
| Error | Compile | **Student 1** | Unknown Identifier |
| Error | Compile | **Student 2** | Unknown Identifier |
| Error | Compile | **Student 1** | Unknown Identifier |
| Technical | Confusion | **Student 1** | i am confused on what is going on |
| Technical | Confusion | **Student 2** | idk im so confused |

**Table 6** Excerpt from a student collaborative dialogue that ends their session with no errors. Each dialogue state was determined automatically using the HMM presented in this work

| State | Action | Student | Text |
|---|---|---|---|
| Success | Compile | **Student 2** | Success |
| Success | Statement | **Student 1** | i chose a random one i saw in the category |
| Edit | Code editing | **Student 2** | .... |
| Success | Compile | **Student 2** | Success |
| Success | Compile | **Student 1** | Success |
| Success | Closing | **Student 1** | there we are done |
| Success | Positive feedback | **Student 2** | the last guitar sounds pretty good |

*satisfaction* tended to focus on solving errors together before moving on to other tasks and explicitly stated when they were moving onto the next task once a task was completed. For example, the excerpt in Table 7, we can see students trying to figure out why their song is not sounding the way they expected. ***Student 1*** suggests a solution, saying "we need to change the track", and ***Student 2*** replies that "... so even if it was on the same track it should work". They both edit the code, and ***Student 2*** comments on a change made by Student 1, saying "if we add it there we would need to add a parameter". They continue to work on their artifact collaboratively, resolving one part of the issue and directing their attention to the next part with ***Student 2*** saying "Alright lets see where the problem is from here" and ***Student 1*** agreeing. The collaborative approach to solving errors from the previous excerpt can be contrasted with the earlier excerpt in Table 5, where ***Student 2*** tries to compile and listen to the song, but receives an error. Neither student mentions anything about the error until ***Student 1*** indirectly references it at

**Table 7** Excerpt from a student collaborative dialogue where a pair focuses on one error before moving on. Each dialogue state was determined automatically using the HMM presented in this work

| State | Action | Student | Text |
|---|---|---|---|
| Technical | Statement | **Student 1** | we need to change the track |
| Technical | Editing code | **Student 1** | .... |
| Technical | Statement | **Student 2** | But it goes from measures 1-8 and then 12-16 so even if it was on the same track it should work |
| Edit | Editing code | **Student 2** | .... |
| Success | Compile | **Student 2** | Success |
| Edit | Editing code | **Both** | .... |
| Error | Compile | **Student 1** | Type Error |
| Edit | Editing code | **Student 1** | .... |
| Error | Statement | **Student 2** | if we add it there we would need to add a parameter |
| Edit | Editing code | **Student 2** | .... |
| Error | Compile | **Student 1** | Type Error |
| Edit | Editing code | **Both** | .... |
| Success | Compile | **Student 2** | Success |
| Edit | Editing code | **Both** | .... |
| Error | Compile | **Student 1** | Type Error |
| Edit | Editing code | **Both** | .... |
| Success | Compile | **Student 2** | Success |
| Technical | Directive | **Student 2** | Alright lets see where the problem is from here |
| Technical | Directive accept | **Student 1** | ok |

the end of the session by saying "i am confused on what is going on" and ***Student 2*** agrees by saying "idk im so confused". The lack of communication about code errors may be because when facing errors in their code, some students see the errors as a negative reflection of their abilities (Kinnunen & Simon, 2010; Gorson et al., 2021). The negative self-assessments related to errors in code could suggest that pairs that engaged in more discussion around errors were more confident in their abilities.

Additionally, expressions of uncertainty or confusion are an important aspect of collaborative dialogue and can provide learning opportunities when resolved (Berlyne, 1978). However, unresolved uncertainty can lead to negative affective states such as boredom and frustration (D'Mello & Graesser, 2012). We found that higher rated pairs usually tried to clear up any confusion about the task or artifact, even if the error had been resolved. For example, in the excerpt in Table 8, ***Student 1*** makes some changes to the code and resolving an error, but ***Student 2*** references the removed code saying "i was trying to make that an effect". ***Student 2*** then makes some changes to the code and is able to successfully compile it. ***Student 1*** apologizes and indicates their confusion saying "my b i just didn't know what was wrong". Despite the problem being resolved and the code continuing to compile successfully after changes from Student 1, ***Student 2*** elaborates on their earlier explanation by replying "your all good, im trying to

**Table 8** Excerpt from a student collaborative dialogue showing a student addressing their partner's confusion and explicitly stating their goal. Each dialogue state was determined automatically using the HMM presented in this work

| State | Action | Student | Text |
|---|---|---|---|
| Error | Compile | **Student 1** | Type Error |
| Edit | Editing code | **Student 1** | .... |
| Success | Compile | **Student 1** | Success |
| Success | Statement | **Student 2** | i was trying to make that an effect |
| Edit | Editing code | **Student 2** | .... |
| Success | Compile | **Student 2** | Success |
| Success | Confusion | **Student 1** | my b i just didn't know what was wrong |
| Success | Compile | **Student 1** | Success |
| Edit | Editing code | **Student 1** | .... |
| Success | Compile | **Student 1** | Success |
| Success | Statement | **Student 2** | your all good, im trying to make the CIARA beat slower by setting an effect to change the tempo |

**Table 9** Excerpt from a student collaborative dialogue of a pair not resolving confusion. Each dialogue state was determined automatically using the HMM presented in this work

| State | Action | Student | Text |
|---|---|---|---|
| Technical | Editing code | **Student 1** | .... |
| Technical | Confusion | **Student 1** | do you know how to do the envelope? |
| Technical | Confusion | **Student 2** | i don't |
| Technical | Editing code | **Student 1** | .... |

make the CIARA beat slower by setting an effect to change the tempo". In contrast, an excerpt in Table 9 shows ***Student 1*** asking "do you know how to do the envelope?" and ***Student 2*** responds that they do not. ***Student 1*** returns to editing code, and the topic is never brought up again. If ***Student 1*** did find the solution, it was not discussed with their partner. This pair went on to only have six more utterances and ended the session with more unresolved expressions of confusion and compile errors. Rodríguez et al. (2017b) compared uncertainty in pair programming for high and low performing pairs based on task solution quality. They found a similar pattern where higher performing pairs tended to address and resolve uncertainty and also focused on resolving a single task before moving on while lower performing pairs did not.

In our examination of the *Compilation Error* state, we find such communication between partners to be an important part of resolving errors, which is consistent with prior research. Teasley (1997) found more types of transactive discussion—in which partners act on each other's reasoning—in pairs and that the pairs generate a deeper understanding of the problem more quickly. Additionally, Murphy et al. (2010) observed improvement in the number of problems

completed and debugging during pair programming with increased communication and certain types of transactive statements. The increased communication observed in the pairs with the highest *combined satisfaction* may have helped the pairs resolve their errors more quickly, thus having fewer interactions in the *Compilation Error* state.

### Curriculum Browsing → Code Editing: Negative Relationship with Combined Satisfaction

The model revealed that more transitions from *Curriculum Browsing* → *Code Editing* had a negative relationship with *combined satisfaction*. While examining the interactions of the pairs, we noted two different ways—targeted and undirected—that students used the curriculum resource. The first type, targeted, was when students had a specific problem they were working on, like resolving an error or writing a custom function, and they went to the specific curriculum resources that addressed their problem. Students using the curriculum documents to address a specific problem is occasionally part of the *Curriculum Browsing* state, but is most often included in one of the other states, such as the *Compilation Error* state. An excerpt of a student receiving a *Type Error* and then accessing the *31.7: Type Error* curriculum document in the *Compilation Error* state can be seen in Table 10.

However, the type of curriculum usage, undirected, that characterizes the *Curriculum Browsing* state, involves students clicking though the curriculum indiscriminately. Kinnunen and Simon (2010) found that when students are not sure how to begin, they would freeze up or start looking through their resources for examples. In some of the pairs with lower *combined satisfaction*, one student seemed to be unsure of how to get started and did little or no coding within the session, and their partner did either all or almost all of the coding. The students that browsed the curriculum and made few contributions also asked questions like "how do I play it?" without any response from their partners. The curriculum browsing behavior usually started from the introductory unit and followed it through the subsequent topics, which is depicted in the excerpt in Table 11. In the same session from the excerpt in Table 11, ***Student 1*** tries to add some code to the project about a quarter of the way through the session. ***Student 2*** says "What are you trying to write?" and deletes Student 1's contribution. ***Student 1*** does not try to add anything else to the project

**Table 10** Excerpt from a student collaborative dialogue that depicts the targeted Curriculum usage as part of the Compilation Error state instead of the Curriculum Browsing state. Each dialogue state was determined automatically using the HMM presented in this work

| State | Action | Student | Text |
|---|---|---|---|
| Edit | Editing code | **Student 1** | .... |
| Error | Compile | **Student 1** | Type Error |
| Error | Curriculum | **Student 1** | 31.7: Type Error |
| Edit | Editing code | **Student 1** | .... |
| Success | Compile | **Student 1** | Success |

**Table 11** Excerpt from a student collaborative dialogue showing a student browsing through basic curriculum documents while their partner works in the code editor. Each dialogue state was determined automatically using the HMM presented in this work

| State | Action | Student | Text |
|---|---|---|---|
| Curriculum | Curriculum | **Student 1** | Unit 1 Introduction |
| Curriculum | Curriculum | **Student 1** | 1.1: Why Learn Programming for Music? |
| Curriculum | Curriculum | **Student 1** | 1.2: Tools of the Trade: DAWs and APIs |
| Curriculum | Curriculum | **Student 1** | 1.3: The EarSketch Workspace |
| Edit | Editing code | **Student 2** | .... |
| Success | Compile | **Student 2** | Success |
| Success | Compile | **Student 1** | Success |
| Edit | Editing code | **Student 2** | .... |
| Curriculum | Curriculum | **Student 1** | 1.4: Running a Script |
| Curriculum | Curriculum | **Student 1** | 1.5: Adding Comments |
| Curriculum | Curriculum | **Student 1** | 1.6: The DAW in Detail |

for the rest of the session, but does go back to access some of the basic curriculum documents again. Chaparro et al. (2005) saw similar patterns where a higher skilled partner would take full control when paired with a more passive and lower skilled partner. We can contrast this behavior with an interaction from a pair with higher *combined satisfaction*. One student clicks through the curriculum at the very beginning of the session then asks their partner *"so we're just making a ringtone"*. They continue to click through the curriculum for a couple of minutes until their partner answers *"yes"* and explains *"we need to figure out how many measures it takes to get to 60 seconds"*. After this explanation from their partner, the student stopped browsing the curriculum and started the project.

Out of the two curriculum usage methods we identified—targeted and undirected—the *Curriculum Browsing* state is characterized by the latter. In the pairs that had a high relative frequency of *Curriculum Browsing* → *Code Editing* transitions, typically one partner browsed the curriculum while the other worked in the code editor, and the contributions to the artifact were unequal. The excerpt previously referenced in Table 11 depicts ***Student 2*** working in the code editor while ***Student 1*** browses the curriculum, creating a transition from *Curriculum Browsing* → *Code Editing*. The negative relationship between the transition from *Curriculum Browsing* → *Code Editing* and *combined satisfaction* suggests that when one student is unsure in a way that causes this undirected curriculum usage, the student may require extra support.

### Aesthetic Dialogue → Technical Dialogue: Negative Relationship with Combined Satisfaction

Previously we observed that the typical progression of a session begins in the *Social Dialogue* state, then pairs typically transition to the *Aesthetic Dialogue* state. Next, pairs move on to the *Technical Dialogue* state, which is followed by the code editing

states (Fig. 5). However, the regression model (Table 4) showed a negative relationship between the relative frequency of the transition from *Aesthetic Dialogue* → *Technical Dialogue* and *combined satisfaction*. To understand why the transition *Aesthetic Dialogue* → *Technical Dialogue* would have a negative relationship with *combined satisfaction*, we will review prior literature including cognitive load and the dynamic between learning and making, then ground this discussion in excerpts from the corpus.

Mastering basic coding skills already creates a large cognitive load on the student, and the addition of a creative component greatly increases the cognitive load (Grover, 2020, p.27). Programming is considered to have a high intrinsic load, which is the difficulty inherent in a task (Fincher and Robins, 2019, p.257). Collaborative learning can help reduce this cognitive load by introducing a *collective working memory*, with communication being essential to its creation (Kirschner et al., 2011; Kirschner et al., 2018). In our corpus, as *Code Editing* and the debugging process increases, dialogue decreases, which can be seen in the ridgeline depicting the frequency of the hidden states over the session in Fig. 5. The decrease in dialogue is more apparent when we examine the pairs with the lowest *combined satisfaction*. However, when we look at the pairs with the highest *combined satisfaction*, dialogue is relatively consistent throughout (Fig. 7). Kirschner et al. (2018) argues that during complex tasks, collaboration works as a scaffold for the knowledge acquisition process of each partner, and if it does not, it can become ineffective by causing too much extraneous load. Seeing a drop in dialogue when coding and debugging begins is unsurprising because Lahtinen et al. (2005) found that the most difficult programming issue for students is *finding bugs in their own programs*. Some possible explanations for observing more consistent dialogue throughout the sessions of the pairs with the highest *combined satisfaction* are that these pairs did not have as much trouble with the programming aspects, like debugging, or that the partnership was effective at reducing some of the cognitive load. In the next three paragraphs we will discuss excerpts from the corpus from a cognitive load perspective.

One difference we observed between the pairs with the highest and lowest *combined satisfaction* was in how the pairs approached the task. The pairs with the highest *combined satisfaction* tended to focus on their task requirements first, preferring to get simpler code elements in place before working out the details of the music. In the excerpt in Table 12 we can see just that—***Student 1*** making suggestions about the sounds they should use, but ***Student 2*** wants to focus on making the loop before playing with the sounds. Breaking the task down into smaller more manageable parts by focusing on the basic coding elements first may help reduce the cognitive load in two ways. First, this resembles the iterative refinement process where the program is built in chunks, includes testing and debugging before adding another level of complexity, and is one of the strategies that supports the debugging process (Grover, 2020, p.262). Second, focusing on the aesthetic and technical concerns separately may keep students from being overburdened with the cognitive load of addressing both components simultaneously.

The iterative refinement process is demonstrated by the pair featured in Table 13, where initially, ***Student 2*** suggests making a function with no arguments and then assigns the individual pieces of the function to each partner. By starting with no

**Table 12** Excerpt from a student collaborative dialogue showing a pair prioritizing code and task requirements before selecting sounds. Each dialogue state was determined automatically using the HMM presented in this work

| State | Action | Student | Text |
|---|---|---|---|
| Aesthetic | Proposal | **Student 1** | we should put like a synth or something to that effect |
| Aesthetic | Proposal | **Student 1** | add some like futuristic noises or airhorns or something |
| Aesthetic | Proposal | **Student 2** | We need to add the stuff to make it a loop first and then play with the sounds |
| Technical | Statement | **Student 2** | so we have the essential stuff done |
| Technical | Acknowledgement | **Student 1** | yeah |
| Technical | Editing code | **Student 2** | .... |
| Technical | Directive | **Student 1** | im gonna put each sound into a variable to make it easier to read |

**Table 13** Excerpt from a student collaborative dialogue of a pair discussing general aesthetic choices before moving onto code requirements and assignments. Each dialogue state was determined automatically using the HMM presented in this work

| State | Action | Student | Text |
|---|---|---|---|
| Social | Social | **Student 1** | hey |
| Social | Social | **Student 2** | hey |
| Aesthetic | Proposal | **Student 1** | rock theme? |
| Aesthetic | Proposal accept | **Student 2** | sounds good |
| Technical | Acknowledgement | **Student 1** | alright |
| Technical | Directive | **Student 2** | lets start with just making a function with no arguments first |
| Technical | Directive accept | **Student 1** | k |
| Technical | Diredctive | **Student 2** | you make the fitMedia |
| Technical | Directive accept | **Student 1** | ok |
| Technical | Directive | **Student 2** | i can do the loop |
| Technical | Directive accept | **Student 1** | alright |

arguments in the function, the pair is able to get the basic elements in place and working before increasing the complexity. Additionally, this pair's dialogue was very consistent throughout the entire session with more of their *Technical Dialogue* occurring at the beginning of the session and more of their *Aesthetic Dialogue* occurring in the third quarter of the session. In comparison, another pair began their session by discussing the aesthetic parts of the project, assigning roles to each partner, but did not follow the iterative refinement process. The pair does not attempt to compile their code until 20 minutes after the session starts when they already had multiple functions written, which may have contributed to the difficulty they had trying to fix the syntax error they were receiving. The entirety of the pair's interactions, starting from about half of the way through the session, can be seen in the excerpt in Table 14, which includes their first attempt to compile during the session. Towards the end ***Student 1*** asks "whats wrong with our functions". The pair did not have a single successful compilation during their entire session.

Because adding an aesthetic component can increase the cognitive load of the programming project, focusing on the aesthetic and technical elements individually may help the pairs reduce their cognitive load. Novices in particular may face increased cognitive load from the technical portion of their programming tasks because they are still working to acquire necessary schemata to handle the technical portion of their projects (Fincher and Robins, 2019, p.258). This compartmentalization of the aesthetic and technical components can be seen in this utterance from a pair with a higher *combined satisfaction*: *"Now that we have a function we can call it multiple times but we can just change the sound"*. After their function is working, they can turn their attention to getting the song to sound the way they want. The benefits can be seen later in the session when they are working on the artifact aesthetic when one student says *"Oh I see the problem... When we try and call the function again the piano tries to start playing again on track 3, but its already playing"*

**Table 14** Excerpt from a student collaborative dialogue pair's entire interactions starting from around half of the way through the session. The pair is unable to resolve coding errors. Each dialogue state was determined automatically using the HMM presented in this work

| State | Action | Student | Text |
|---|---|---|---|
| Technical | Editing code | **Student 1** | .... |
| Technical | Statement | **Student 2** | iv'e only found two sounds to use so far |
| Technical | Editing code | **Student 1** | .... |
| Technical | Directive | **Student 1** | u can put them if you want |
| Technical | Editing code | **Both** | .... |
| Technical | Confusion | **Student 2** | do i make another function |
| Technical | Statement | **Student 1** | yeah |
| Edit | Editing code | **Both** | .... |
| Error | Compile | **Student 2** | SyntaxError |
| Edit | Editing code | **Both** | .... |
| Error | Compile | **Student 1** | SyntaxError |
| Technical | Directive | **Student 1** | can you fix the mistake |
| Technical | Directive | **Student 1** | on line 20 and 21 |
| Technical | Directive | **Student 1** | i mean 19 and 20 |
| Edit | Editing code | **Student 2** | ... |
| Error | Compile | **Student 2** | SyntaxError |
| Error | Compile | **Student 1** | SyntaxError |
| Error | Confusion | **Student 1** | whats wrong with our functions |
| Edit | Editing code | **Student 1** | .... |
| Error | Confusion | **Student 2** | i dont know whats wrong |
| Edit | Editing code | **Student 1** | .... |
| Error | Compile | **Student 1** | SyntaxError |

to which their partner replies *"we need to add the thing that changes the track"*. While this still involves changes to the code, being able to compile their code lets them focus on how to change the sound without having to deal with coding errors. As noted previously, when students enter the *Compilation Error* state, they tend to focus on resolving these errors instead of discussing the artifact aesthetics.

While this separation of *Aesthetic Dialogue* and *Technical Dialogue* seems to work for pairs that focus on the technical first, it may not work as well for pairs initially focusing on the aesthetics. In Table 15 we can see a pair listening to and deciding what sounds to use. One partner suggests making a loop, but the other partner says they should get all the instruments and then make it a song. When they do try to move onto making it a song towards the end of the session, ***Student 2*** says *"start finding other instruments, i will turn this part into a base"*. Afterwards it looks like ***Student 1*** finds a sound and says *"try it out now"*, but they do not get a chance to discuss it as the session ends a minute later. This pair seems to have gotten stuck on the aesthetics and were unable to progress to putting the song together. This evokes another critique of creative coding—that it can move

**Table 15** Excerpt from a student collaborative dialogue of a pair deciding sounds before getting code elements in place. Each dialogue state was determined automatically using the HMM presented in this work

| State | Action | Student | Text |
|---|---|---|---|
| Success | Compile | **Student 1** | Success |
| Success | Pos feedback | **Student 1** | Woah thats neat |
| Aesthetic | Proposal | **Student 2** | do you want to use those drums |
| Aesthetic | Proposal accept | **Student 1** | u can put them if you want |
| Aesthetic | Acknowledgement | **Student 2** | ok |
| Aesthetic | Proposal | **Student 1** | so like do you want to make a loop |
| Aesthetic | Proposal reject | **Student 2** | not yet |
| Social | Social | **Student 1** | Wait nevermind |
| Social | Social | **Student 1** | oh |
| Technical | Directive | **Student 2** | First lets just get all the instruments then we make it a song |

the focus from learning to making (Grover, 2020, p.26), creating what has been referred to as the 'virtuous' and 'vicious' cycles of STEAM CS education (Moore et al., 2017). The 'virtuous' cycle involves being motivated by the aesthetic part of the artifact to learn more advanced coding concepts to continue improving the artifact. The 'vicious' cycle happens when students learn just enough to be able to make the artifact and focus almost entirely on the artifact aesthetics and do not progress their computational learning. In the previous example, the students knew enough to be able to play the sounds they picked, but did not use custom functions, loops, or even vary the measures of the sound clips—all clips started in measure 1 and ended at 11—to turn the picked sounds into a song.

When students put too much focus on the aesthetic without having a solid foundation of the fundamentals of programming, students' aesthetic ambitions can exceed their technical capabilities (Grover, 2020, p.27). While some challenges can motivate students to put more effort into the activity, unresolved frustration can lead to boredom and disengagement (D'Mello & Graesser, 2012). One reason the higher rated pairs may benefit from focusing on the technical aspects first is that it may keep students' aesthetic ambitions more realistic for their programming proficiency. This can keep the challenges students face while working on their artifact motivating instead of frustrating. One pair with lower *combined satisfaction* in the excerpt in Table 16, starts the session heavily focusing on the aesthetic details, even suggesting making a jingle for a brand. While they have some success early on, they begin running into more errors as they get into the final quarter of the session. Eventually ***Student 2*** says *"it looks fine. i don't know why its not working. we should look back into unit one"* and begins looking through the curriculum. They are able to get some successful compiles by removing code, but they end the session with a *TypeError* after adding more code in. Murphy et al. (2010) found similar behavior when pairs became stuck during debugging.

**Table 16** Excerpt from a student collaborative dialogue of a pair focusing on the aesthetic details of the song. Each dialogue state was determined automatically using the HMM presented in this work

| State | Action | Student | Text |
|---|---|---|---|
| Aesthetic | Passing Responsibility | **Student 1** | what kind of song should we do |
| Aesthetic | Passing Responsibility | **Student 2** | Do you want to use a song with a lot of beats or slower tempo? |
| Technical | Statement | **Student 1** | I normally use the same tempo |
| Technical | Confusion | **Student 2** | what type is that? |
| Technical | Statement | **Student 1** | the same 120 |
| Aesthetic | Passing Responsibility | **Student 2** | okay what sounds do you want to use? |
| Aesthetic | Proposal | **Student 2** | we could start with some percussion to have a beat to start off |
| Aesthetic | Proposal Accept | **Student 1** | that sounds fine with me |
| Aesthetic | Proposal | **Student 2** | do you want some type of drum or piano? |
| Aesthetic | Proposal Accept | **Student 1** | drums sound good |
| Aesthetic | Passing Responsibility | **Student 1** | what's our theme tho |
| Aesthetic | Editing Code | **Student 1** | .... |
| Aesthetic | Confusion | **Student 2** | i don't know |
| Aesthetic | Proposal | **Student 2** | what about a jingle for a brand |
| Technical | Statement | **Student 1** | We'll figure that out later then |

Focusing on making sure the code works first can also ensure the pairs are able to work on the aesthetic portions of their song. This is because if the code will not compile, the students cannot hear their song. The excerpt referenced previously in Table 5 shows one student ask their partner if they want to make any changes to the song, and the other student says they need to listen to it first. They are unable to compile the code and thus unable to listen to the song they created. Then they spend the rest of their session debugging. Encountering technical errors when trying to work out the aesthetic aspects of the artifact can force students to focus on the error and potentially frustrate students that would prefer to work on the artifact aesthetics. In contrast, a pair previously referenced finished their function first and was then able to focus on the aesthetic aspects of their artifact like changing the track. This may be why the pairs that tended to get the basic code elements working first had higher *combined satisfaction*.

In this section, we discussed cognitive load and the dynamic between learning and making as possible reasons for the negative relationship between the transition from *Aesthetic Dialogue* → *Technical Dialogue*. Using excerpts from our corpus, we highlighted some of the differences between the pairs with the highest and lowest *combined satisfaction*, which include use of iterative refinement and focusing on task requirements first. We noted that the pairs with the highest *combined satisfaction* had more consistent dialogue throughout the session, including during the debugging phase when dialogue decreased considerably between other pairs. Some reasons for the higher levels of dialogue during the debugging phase could be that the pairs had less trouble with the programming aspects, like debugging, or that

the partnership was effective at reducing some of the cognitive load. Additionally, focusing on task requirements first seemed to allow the pairs that did this to put more focus on the aesthetic elements in the later half of the session, whereas several of the pairs that focused heavily on aesthetics early tended to spend the later half of the session dealing with compile errors. If students are more motivated to discuss problems involving the aesthetic elements rather than compile errors, it could explain some of the increased dialogue in the pairs with higher *combined satisfaction* since they tended to be able to focus on aesthetic problems in the later half of the session. Frustration with debugging and a desire to work on the aesthetic portions may also explain some of the virtuous/viscous cycle discussed previously.

### Implications for Intelligent System Support

The findings of this research, while preliminary, provide insights into how co-creative processes are associated with *combined satisfaction*, and they illuminate potential opportunities around compilation errors, uncertainty, and the dynamic between learning and making to support collaboration during co-creative learning tasks. Some of these findings are specific to the co-creative process, while others apply to collaboration in general, and thus, have important implications for both co-creative agents and collaborative support systems. We will discuss each finding with its implications for both co-creative agents and collaborative support systems, but first we will define and briefly discuss the role of these two types of intelligent systems.

Co-creative agents fulfill the role of collaborator on open-ended creative tasks. Using agents as partners to human learners has demonstrated benefits including significantly higher levels of shared understanding, progress monitoring, and feedback (Rosen, 2015). Applications of co-creative agents range from improving collaborative skills, like encouraging "deep thinking" and initiative taking (Howard et al., 2017), to providing emotional support for elementary school students learning to code (Morales-Urrutia et al., 2020). A co-creative agent can support human creativity through supporting the creation of music through code (Truesdell et al., 2021) and supporting collaborative ideation through sketches (Lin et al., 2020).

Collaborative support systems are systems where the primary role is to support collaboration between humans. Some of these systems perform functions like encouraging productive dialogue during collaborative learning (Dyke et al., 2013; Tegos et al., 2014), while others help students regulate their collaboration by prompting students to use different reflection strategies (Sankaranarayanan et al., 2020).

### Support for Error Resolution

Our findings indicate that the pairs that had more interactions in the *Compilation Error* state tended to rate their partners lower than pairs that had fewer interactions in the *Compilation Error* state. We identified patterns that were common in the higher rated pairs that were frequently missing in the lower rated pairs: they tended to have more of their errors resolved by the end of the session, focused on a single task together, explicitly stated when they were done and ready to move on, and

worked together to resolve uncertainty even if the coding error had already been resolved. We can use these observations from more successful pairs to guide intelligent system support.

Because dialogue was such an effective tool for resolving errors, if co-creative agents detect an extended amount of time spent in the *Compilation Error* state, they could try to engage the student in discussion about the error and task. If the student is still unable to resolve the issue after talking through the issue, the co-creative agent could provide extra help resolving programming errors or offer examples of working code so that the student can remain engaged with the aesthetic aspects of the work.

For collaborative support systems, support could look like ensuring both partners are working on the same problem if there is an extended amount of time spent in the *Compilation Error* state. That could involve prompting partners to explicitly state what they believe the problem is and encouraging them come up with a plan for how they will fix it. Additionally, after the error is resolved, the system might have partners confirm with each other what the issue was and what fixed the error. This would give both students the opportunity to benefit from the learning experience.

### Support for Help Seeking

The behavior of using curriculum documents in the way that characterizes the *Curriculum Browsing* state—where a student clicks through many different topics instead of seeking out a specific topic—can potentially identify when a student is having trouble getting the help they need. However, our findings show that lower *combined satisfaction* was only associated with the relative frequency of transitions from *Curriculum Browsing* → *Code Editing*, so this browsing behavior itself does not necessarily indicate issues in the collaboration. This transition was most common when one student would work in the code editor while their partner browsed the curriculum. In these situations, contributions to the artifact were very unequal and conversation was limited, and in many cases the editing student ignored questions from the browsing student.

While a co-creative agent would not likely be designed to work on an artifact without input and contributions from the human partner, there are still things to be learned from this finding. If an agent notices this type of curriculum browsing behavior, they could try to find out what the student is having trouble with and potentially offer resources, ideas, or encouragement. For example, if the problem is related to the aesthetic aspects, like how to decide on a genre, the agent can help the student brainstorm or give examples of different sounds. At a minimum, co-creative agents should treat this curriculum browsing behavior as a indicator to check in with the student.

Alternatively, when collaborative support systems notice this behavior, it could indicate problems with collaboration between the partners. Systems can encourage the browsing partner to communicate any uncertainty they have with their partner. Systems could also monitor chat and artifact contributions. If the browsing partner has tried to communicate, but the editing partner has not responded, the system could give a nudge or reminder to draw the attention of the editing partner to the

browsing partner's message. If the browsing partner has not made many contributions to the artifact, it could encourage the pair to set specific tasks for each partner. In extreme cases, like where one partner ignores the other and possibly even removes the browsing partner's contributions, an alert to the teacher about the issue may be warranted so they can check in on the pair.

### Scaffolding Aesthetic and Technical Elements

Prior research has shown that when students talk through what they want to make first, it prevents students from putting too much emphasis on project requirements and not enough on personal relevance (Grover, 2020, p.23). Additionally, many students can be more motivated by the personally relevant aspects of the project, like creating music, than the programming itself (Magerko et al., 2016). Furthermore, when many of the aesthetic decisions of a project are left to the end, they can require major changes to the technical aspects. In EarSketch, this may look like changing or adding new code elements to add a drum fill effect to a song. This can potentially lead to sticking with an aesthetic decision that is not preferred to avoid rewriting the program or being unable to finish the changes required if students run out of time to work on their artifact. However, despite the benefits of including aesthetic elements in programming, our findings suggest that students could likely use more support during the co-creative process in two main areas—reducing cognitive load and scaffolding the aesthetic process.

For agents acting as co-creative partners, students can benefit from the knowledge scaffolding provided by collaboration described by Kirschner et al. (2018) by having a partner that can provide the right knowledge at the appropriate time. For example, providing relevant worked or partial worked examples (Zhi et al., 2019; McLaren & Isotani, 2011) and encouraging good coding and debugging habits as they move through the project are ways that can reduce the cognitive load of the technical portion of the task and allow students to spend more time on creating. To help with the aesthetic elements, co-creative agents could use details from the artifact and make suggestions or give examples for moving what the student already has done towards a completed version of the artifact. For example, if students are just inserting sounds without progressing to a song-like structure, agents could suggest a custom function to make a chorus with sounds already present in the code editor or even provide one. This could give students a clear direction for moving forward on completing their artifact.

Collaboration support systems can identify when pairs encounter issues through dialogue features (Goodman et al., 2005), such as dialogue decreasing when errors increase, and encourage partners to communicate. They can even help guide the conversation when pairs are not having productive conversation (Dyke et al., 2013). These systems can also encourage pairs to set explicit goals or tasks—like using a custom function—and help them track which ones they've accomplished. Similar to the co-creative agent, these systems could also suggest more advanced code structures if the pair seems not to be making progress towards completing the artifact.

## Conclusion

Collaborative learning has many benefits, ranging from academic achievement to psychological well-being. In the context of computer science education, this means better code, higher retention rates, and more confidence (Braught et al., 2011; McDowell et al., 2002). A key component of collaboration is collaborative dialogue, a complex exchange of ideas where speakers can work together to solve problems and exchange knowledge. The complexity continues to increase in creative domains which require co-creative dialogue, such as creative coding. Understanding co-creative dialogues are a key step to furthering our understanding of co-creative learning and our ability to design intelligent systems to support it.

To understand the interactions of learners during co-creative tasks, 68 dyads of high school students in separate spaces performed collaborative coding tasks in the EarSketch learning environment. The tasks involved coding short segments of music, such as a 30 second ringtone. We collected their textual dialogue and the actions performed in the EarSketch interface and used these to learn a seven state hidden Markov model. The model has four hidden states primarily composed of interactions within the interface and three hidden states primarily composed of dialogue. We then used post-survey partner rating scores to identify the hidden states or their transitions that may impact these ratings. We found that the higher the relative frequency of interactions in the *Compilation Error* state and the higher the relative frequency of transitions from *Aesthetic Dialogue* → *Technical Dialogue* state and *Browsing Curriculum* → *Code Editing* state, the lower the partner satisfaction.

In this study we identified a common overall flow for the sessions where they typically begin in the *Social Dialogue* state, moving onto the *Aesthetic Dialogue* state after that, followed by the *Technical Dialogue* state before moving into a heavier coding and debugging phase that continues throughout the session. We also identified how the types of dialogue differ based on whether the pairs are transitioning out of the *Compilation Success* or the *Compilation Error* state. The *Compilation Success* state seems to be a point where pairs can renegotiate the aesthetic or technical details of an artifact whereas the *Compilation Error* state typically leads to pairs discussing the technical details of an artifact. Based on the relationship of the hidden states and their transitions with *combined satisfaction*, we identified three main areas where students may need support in these creative coding contexts. pairs that are spending a lot of time in the *Compilation Error* state may need help with error resolution, which can involve encouraging or structuring dialogue between partners. When one or more partners is using the curriculum in the way we see during the *Curriculum Browsing* state, this can indicate that students may be struggling with a broad problem like how to start a task, and if the partner is spending a lot of time in the code editor, potentially an issue with the collaboration itself. Providing project-specific resources and encouraging dialogue can be used to help students get started on a task and answer general task questions they may have. Finally, reducing cognitive load and scaffolding the aesthetic and technical processes may be one way to help students manage the dynamic between learning and making.

This work models the co-creative process and explores the relationship between the model and partner satisfaction. Despite its exploratory nature, this work identifies potential opportunities for support during co-creative learning, with some of the findings being unique to creative domains (e.g., managing the aesthetic and technical portions of a project) while the others are more broadly applicable to collaborative coding. However, there are several limitations to note. Because our findings on the relationship between the hidden states and their transitions and *partner satisfaction* are correlational, we are cannot establish causality. Additionally, because we only have pre-survey data for 44 of the 136 students, we were unable to investigate the impact of the students' own prior experience, confidence, and enjoyment for both computing and music on these findings. Further, this work does not consider any measures of task performance as outcomes. The only indication of task success or failure is the result of compiling the code (success or error) included as an observation symbol in the hidden Markov model. Nevertheless, this study adds to our understanding of co-creative processes and illuminates several possible directions for future research. More research is needed into how to best support students as they balance aesthetic and technical concerns during co-creative learning. Additionally, as this research only focused on partner satisfaction as outcomes, research is still needed into how co-creative processes impact artifact quality or learning outcomes as well as investigating how prior experience and attitudes may influence these outcomes. Another possible research direction for future work would be investigating how various types of feedback, such as playing the created music, impacts the co-creative process and partner satisfaction. Intelligent systems are uniquely positioned to utilize this research to support students during co-creative learning, by either identifying and encouraging good co-creative behaviors or by being good co-creative partners themselves.

**Acknowledgements** This material is based upon work supported by the National Science Foundation under grants DRL-1813740 and DRL-1814083. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

**Funding** This research was funded by the National Science Foundation, DRL-1813740 and DRL-1814083.

## Declarations

**Ethics approval** This work was approved by the University of Florida's IRB prior to the conducted research.

**Consent to participate** ThisEach student represented in this data assented and their parents consented for them to be part of this work.

**Consent for Publication** All authors have approved the manuscript and agree with its submission to the International Journal of Artificial Intelligence in Education.

**Conflict of Interests** There were no conflicts of interest to report for this research.

## References


Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, *19*(6), 716–723. https://doi.org/10.1109/TAC.1974.1100705.

Arroyo, I., Wixon, N., Allessio, D., Woolf, B., Muldner, K., & Burleson, W. (2017). Collaboration improves student interest in online tutoring. Artificial intelligence in education, pp. 28–39. https://doi.org/10.1007/978-3-319-61425-0_3.

Bales, R.F., & Strodtbeck, F.L. (1951). Phases in group problemsolving. *The Journal of Abnormal and Social Psychology*, *46*(4), 485. https://doi.org/10.1037/h0059886.

Berlyne, D.E. (1978). Curiosity and learning. *Motivation and emotion*, *2*(2), 97–175. https://doi.org/10.1007/BF00993037.

Boyer, K.E., Ha, E.Y., Wallis, M.D., Phillips, R., Vouk, M.A., & Lester, J.C. (2009). Discovering tutorial dialogue strategies with hidden markov models. https://doi.org/10.3233/978-1-60750-028-5-141.

Boyer, K.E., Phillips, R., Ingram, A., Ha, E.Y., Wallis, M.D., Vouk, M.A., & Lester, J.C. (2011). Investigating the relationship between dialogue structure and tutoring effectiveness: a hidden markov modeling approach. *International Journal of Artificial Intelligence in Education*, *21*(1), 65–81. https://doi.org/10.3233/JAI-2011-018.

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. https://doi.org/10.1145/1921607.1921609.

Campe, S., Denner, J., Green, E., & Torres, D. (2020). Pair programming in middle school: variations in interactions and behaviors. *Computer Science Education*, *30*(1), 22–46. https://doi.org/10.1080/08993408.2019.1648119.

Carpenter, D., Emerson, A., Mott, B.W., Saleh, A., Glazewski, K.D., Hmelo-Silver, C.E., & Lester, J.C. (2020). Detecting off-task behavior from student dialogue in game-based collaborative learning. Artificial intelligence in education, pp. 55–66. https://doi.org/10.1007/978-3-030-52237-7_5.

Chaparro, E.A., Yuksel, A., Romero, P., & Bryant, S. (2005). Factors affecting the perceived effectiveness of pair programming in higher education. In *Proceedings of the 17th workshop of the psychology of programming interest group* (pp. 5–18).

Chng, E., Seyam, M.R., Yao, W., & Schneider, B. (2020). Using motion sensors to understand collaborative interactions in digital fabrication labs. Artificial intelligence in education, pp. 118–128. https://doi.org/10.1007/978-3-030-52237-7_10.

Davidson, N., & Major, C.H. (2014). Boundary crossings: Cooperative learning, collaborative learning, and problem-based learning. *Journal on Excellence in College Teaching*, *25*(3/4), 7–55.

Denner, J., Werner, L., Campe, S., & Ortiz, E. (2014). Pair programming: Under what conditions is it advantageous for middle school students? *Journal of Research on Technology in Education*, *46*(3), 277–296. https://doi.org/10.1080/15391523.2014.888272.

Dich, Y., Reilly, J., & Schneider, B. (2018). Using physiological synchrony as an indicator of collaboration quality, task performance and learning. Artificial intelligence in education, pp. 98–110. https://doi.org/10.1007/978-3-319-93843-1_8.

Dillenbourg, P. (1999). What do you mean by'collaborative learning'?. In P. Dillenbourg (Ed.) *Collaborative learning: Cognitive and computational approaches*. Oxford: Elsevier.

D'Mello, S., & Graesser, A. (2012). Dynamics of affective states during complex learning. *Learning and Instruction*, *22* (2), 145–157. https://doi.org/10.1016/j.learninstruc.2011.10.001.

Dyke, G., Adamson, D., Howley, I., & Rose, C.P. (2013). Enhancing scientific reasoning and discussion with conversational agents. *IEEE Transactions on Learning Technologies*, *6*(3), 240–247. https://doi.org/10.1109/TLT.2013.25.

Ferschke, O., Yang, D., Tomar, G., & Rosé, C.P. (2015). Positive impact of collaborative chat participation in an edx mooc. In C. Conati, N. Heffernan, A. Mitrovic, & M.F. Verdejo (Eds.) *Artificial intelligence in education*. https://doi.org/10.1007/978-3-319-19773-912 (pp. 115–124).

Fincher, S.A., & Robins, A.V. (2019). The cambridge handbook of computing education research. Cambridge University Press.

Freeman, J., Magerko, B., McKlin, T., Reilly, M., Permar, J., Summers, C., & Fruchter, E. (2014). Engaging underrepresented groups in high school introductory computing through computational remixing with earsketch. pp. 85–90. https://doi.org/10.1145/2538862.2538906.

Freeman, J., Magerko, B., & Verdin, R. (2015). Earsketch: a web-based environment for teaching introductory computer science through music remixing. In *The 46th acm technical symposium on computer science education*. https://doi.org/10.1145/2676723.2691869 (p. 5).

Fuller, D., & Magerko, B. (2010). Shared mental models in improvisational performance. In *Proceedings of the intelligent narrative technologies iii workshop*. https://doi.org/10.1145/1822309.1822324. New York, NY, USA: Association for Computing Machinery.

Fuller, D., & Magerko, B. (2011). Shared mental models in improvisational theatre. In *Proceedings of the 8th acm conference on creativity and cognition*. https://doi.org/10.1145/2069618.2069663 (pp. 269–278). New York, NY, USA: Association for Computing Machinery.

Glăveanu, V.-P. (2011). How are we creative together? comparing sociocognitive and sociocultural answers. *Theory & Psychology*, *21*(4), 473–492. https://doi.org/10.1177/0959354310372152.

Gokhale, A.A. (1995). Collaborative learning enhances critical thinking. *Journal of Technology Education*, *7*(1), 22–30. https://doi.org/10.21061/jte.v7i1.a.2.

Goodman, B.A., Linton, F.N., Gaimari, R.D., Hitzeman, J.M., Ross, H.J., & Zarrella, G. (2005). Using dialogue features to predict trouble during collaborative learning. *User Modeling and User-Adapted Interaction*, *15*(1), 85–134. https://doi.org/10.1007/s11257-004-5269-x.

Gorson, J., LaGrassa, N., Hu, C.H., Lee, E., Robinson, A.M., & O'Rourke, E. (2021). An approach for detecting student perceptions of the programming experience from interaction log data. In I. Roll, D. McNamara, S. Sosnovsky, R. Luckin, & V. Dimitrova (Eds.) *Artificial intelligence in education*. https://doi.org/10.1007/978-3-030-78292-4_13 (pp. 150–164).

Graesser, A.C., Fiore, S.M., Greiff, S., Andrews-Todd, J., Foltz, P.W., & Hesse, F.W. (2018). Advancing the science of collaborative problem solving. *Psychological Science in the Public Interest*, *19*(2), 59–92. https://doi.org/10.1177/1529100618808244.

Grover, S. (2020). Computer science in k-12: an az handbook on teaching programming (S. Grover, Ed.). Efinity.

Howard, C., Jordan, P., Di Eugenio, B., & Katz, S. (2017). Shifting the load: a peer dialogue agent that encourages its human collaborator to contribute more to problem solving. *International Journal of Artificial Intelligence in Education*, *27*(1), 101–129. https://doi.org/10.1007/s40593-015-0071-y.

K–12 Computer Science Framework. (2016). K-12 computer science framework. Retrieved from https://k12cs.org/.

Kantosalo, A., Toivanen, J., Xiao, P., & Toivonen, H. (2014). From isolation to involvement: Adapting machine creativity software to support human-computer co-creation. *The fifth international conference on computational creativity*, *2014*, 1–7.

Katuka, G.A., Bex, R.T., Celepkolu, M., Boyer, K.E., Wiebe, E., Mott, B., & Lester, J. (2021). My partner was a good partner: Investigating the relationship between dialogue acts and satisfaction among middle school computer science learners. In *Proceedings of the 14th international conference on computer-supported collaborative learning-cscl 2021*.

Katuka, G.A., Webber, A.R., Wiggins, J.B., Boyer, K.E., Magerko, B., McKlin, T., & Freeman, J. (2022). The relationship between co-creative dialogue and high school learners' satisfaction with their collaborator in computational music remixing. Proc. ACM Hum.-Comput. Interact., 6(CSCW1). https://doi.org/10.1145/3512970.

Kinnunen, P., & Simon, B. (2010). Experiencing programming assignments in cs1: The emotional toll. In *Proceedings of the sixth international workshop on computing education research*. https://doi.org/10.1145/1839594.1839609 (pp. 77–86).

Kirschner, F., Paas, F., & Kirschner, P.A. (2011). Task complexity as a driver for collaborative learning efficiency: The collective working-memory effect. *Applied Cognitive Psychology*, *25*(4), 615–624. https://doi.org/10.1002/acp.1730.

Kirschner, P.A., Sweller, J., Kirschner, F., & Zambrano, J. (2018). From cognitive load theory to collaborative cognitive load theory. *International Journal of Computer-Supported Collaborative Learning*, *13*(2), 213–233. https://doi.org/10.1007/s11412-018-9277-y.

Knobelsdorf, M., & Romeike, R. (2008). Creativity as a pathway to computer science. *SIGCSE Bull.*, *40*(3), 286–290. https://doi.org/10.1145/1597849.1384347.

Lahtinen, E., Ala-Mutka, K., & Järvinen, H.-M. (2005). A study of the difficulties of novice programmers. In *Proceedings of the 10th annual sigcse conference on innovation and technology in computer science education* (pp. 14–18).

Landis, J.R., & Koch, G.G. (1977). The measurement of observer agreement for categorical data. *Biometrics*, *33*(1), 159–174. https://doi.org/10.2307/2529310.

Lin, Y., Guo, J., Chen, Y., Yao, C., & Ying, F. (2020). It is your turn: Collaborative ideation with a co-creative robot through sketch. In *Proceedings of the 2020 chi conference on human factors in computing systems*. https://doi.org/10.1145/3313831.3376258 (pp. 1–14).

Magerko, B., Freeman, J., Mcklin, T., Reilly, M., Livingston, E., Mccoid, S., & Crews-Brown, A. (2016). Earsketch: A steam-based approach for underrepresented populations in high school computer science education. ACM Trans. Computers in Education, 16(4). https://doi.org/10.1145/2886418.

Matsumae, A., Raharja, F.T., Ehkirch, Q., & Nagai, Y. (2021). How the cocreative process affects concept formation. *Proceedings of the Design Society*, *1*, 1775–1786. https://doi.org/10.1017/pds.2021.439.

McDowell, C., Werner, L., Bullock, H., & Fernald, J. (2002). The effects of pairprogramming on performance in an introductory programming course. In *Proceedings of the 33rd sigcse technical symposium on computer science education*. https://doi.org/10.1145/563340.563353 (pp. 38–42).

McLaren, B.M., & Isotani, S. (2011). When is it best to learn with all worked examples?. In G. Biswas, S. Bull, J. Kay, & A. Mitrovic (Eds.) *Artificial intelligence in education*. https://doi.org/10.1007/978-3-642-21869-9_30(pp. 222–229).

Moore, R., Helms, M., & Freeman, J. (2017). Steam-based interventions in computer science: Understanding feedback loops in the classroom. 2017 asee annual conference & exposition. https://doi.org/10.18260/1-2--28842.

Morales-Urrutia, E.K., Ocaña Ch., J.M., Pérez-Marín, D., & Pizarro-Romero, C. (2020). Promoting learning and satisfaction of children when interacting with an emotional companion to program. In I.I. Bittencourt, M. Cukurova, K. Muldner, R. Luckin, & E. Millán (Eds.) *Artificial intelligence in ed- ucation*. https://doi.org/10.1007/978-3-030-52240-7_40 (pp. 220–223).

Murphy, L., Fitzgerald, S., Hanks, B., & McCauley, R. (2010). Pair debugging: a transactive discourse analysis. In *Proceedings of the sixth international workshop on computing education research*. https://doi.org/10.1145/1839594.1839604(pp. 51–58).

Neath, A.A., & Cavanaugh, J.E. (2012). The bayesian information criterion: background, derivation, and applications. *Wiley Interdisciplinary Reviews: Computational Statistics*, *4*(2), 199–203.

Ogan, A., Finkelstein, S., Walker, E., Carlson, R., & Cassell, J. (2012). Rudeness and rapport: Insults and learning gains in peer tutoring. In S.A. Cerri, W.J. Clancey, G. Papadourakis, & K. Panourgia (Eds.) *Intelligent tutoring systems*. https://doi.org/10.1007/978-3-642-30950-2_2 (pp. 11–21).

Rabiner, L., & Juang, B. (1986). An introduction to hidden markov models. *IEEE ASSP Magazine*, *3*(1), 4–16. https://doi.org/10.1109/MASSP.1986.1165342.

Radu, I., Tu, E., & Schneider, B. (2020). Relationships between body postures and collaborative learning states in an augmented reality study. In I.I. Bittencourt, M. Cukurova, K. Muldner, R. Luckin, & E. Millán (Eds.) *Artificial intelligence in education*. https://doi.org/10.1007/978-3-030-52240-7_47 (pp. 257–262).

Rodríguez, F.J., & Boyer, K.E. (2015). Discovering individual and collaborative problem-solving modes with hidden markov models. In C. Conati, N. Heffernan, A. Mitrovic, & M.F. Verdejo (Eds.) *Artificial intelligence in education*. https://doi.org/10.1007/978-3-319-19773-9_41 (pp. 408–418).

Rodríguez, F.J., Price, K.M., & Boyer, K.E. (2017). Exploring the pair programming process: Characteristics of effective collaboration. In *Proceedings of the 2017 acm sigcse technical symposium on computer science education*. https://doi.org/10.1145/3017680.3017748 (pp. 507–512).

Rodríguez, F.J., Price, K.M., & Boyer, K.E. (2017). *Expressing and addressing uncertainty: A study of collaborative problem-solving dialogues. Proceedings of the 12th international conference on computer supported collaborative learning (cscl)*. Philadelphia, PA: International Society of the Learning Sciences.

Roschelle, J. (1992). Learning by collaborating: Convergent conceptual change. *Journal of the Learning Sciences*, *2*(3), 235–276. https://doi.org/10.1207/s15327809jls0203_1.

Roschelle, J., & Teasley, S.D. (1995). The construction of shared knowledge in collaborative problem solving. In C. O'Malley (Ed.) *Computer supported collaborative learning*. https://doi.org/10.1007/978-3-642-85098-1_5 (pp. 69–97).

Rosen, Y. (2015). Computer-based assessment of collaborative problem solving: Exploring the feasibility of human-to-agent approach. *International Journal of Artificial Intelligence in Education*, *25*(3), 380–406. https://doi.org/10.1007/s40593-015-0042-3.

Samoilescu, R.-F., Dascalu, M., Sirbu, M.-D., Trausan-Matu, S., & Crossley, S.A. (2019). Modeling collaboration in online conversations using time series analysis and dialogism. In S. Isotani, E. Millán, A. Ogan, P. Hastings, B. McLaren, & R. Luckin (Eds.) *Artificial intelligence in education*. https://doi.org/10.1007/978-3-030-23204-7_38 (pp. 458–468).

Sankaranarayanan, S., Kandimalla, S.R., Hasan, S., An, H., Bogart, C., Murray, R.C., & Rosé, C. (2020). Agent-in-the-loop: Conversational agent support in service of reflection for learning

during collaborative programming. In I.I. Bittencourt, M. Cukurova, K. Muldner, R. Luckin, & E. Millán (Eds.) *Artificial intelligence in education*. https://doi.org/10.1007/978-3-030-52240-750(pp. 273–278).

SAS Institute Inc. (2020). Jmp®15 fitting linear models [Computer software manual] Cary NC, SAS Institute Inc.

SAS Institute Inc. (2021). JMP®Pro 15. Retrieved from https://www.jmp.com/enus/software/predictive-analyticssoftware.html.

Schneider, B., & Pea, R. (2014). Toward collaboration sensing. *International Journal of Computer-Supported Collaborative Learning*, *9* (4), 371–395. https://doi.org/10.1007/s11412-014-9202-y.

Snyder, C., Hutchins, N.M., Biswas, G., Emara, M., Yett, B., & Mishra, S. (2020). Understanding collaborative question posing during computational modeling in science. In I.I. Bittencourt, M. Cukurova, K. Muldner, R. Luckin, & E. Millán (Eds.) *Artificial intelligence in education*. https://doi.org/10.1007/978-3-030-52240-7_54 (pp. 296–300).

Soller, A. (2001). Supporting social interaction in an intelligent collaborative learning system. *International Journal of Artificial Intelligence in Education*, *12*, 40–62.

Soller, A., & Lesgold, A. (2007). Modeling the process of collaborative learning. In H.U. Hoppe, H. Ogata, & A. Soller (Eds.) *The role of technology in cscl: Studies in technology enhanced collaborative learning*. https://doi.org/10.1007/978-0-387-71136-2_5 (pp. 63–86).

Swain, M. (2000). The output hypothesis and beyond: Mediating acquisition through collaborative dialogue. In *Sociocultural theory and second language learning*, (Vol. 78 pp. 97–114). Oxford University Press.

Teasley, S.D. (1997). Talking about reasoning: How important is the peer in peer collaboration?. In L.B. Resnick, R. Säljö, C. Pontecorvo, & B. Burge (Eds.) *Discourse, tools and reasoning: Essays on situated cognition*. https://doi.org/10.1007/978-3-662-03362-3_16 (pp. 361–384).

Tegos, S., Demetriadis, S., & Tsiatsos, T. (2014). A configurable conversational agent to trigger students' productive dialogue: a pilot study in the call domain. *International Journal of Artificial Intelligence in Education*, *24*(1), 62–91. https://doi.org/10.1007/s40593-013-0007-3.

Truesdell, E., Smith, J., Mathew, S., McKlin, T., Katuka, G.A., Griffith, A.E., & Boyer, K.E. (2021). Supporting computational music remixing with a co-creative learning companion. In *Proceedings of the 12th international conference on computational creativity* (pp. 113–121).

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. https://doi.org/10.1016/j.ijcci.2018.03.001.

Tsan, J., Rodríguez, F.J., Boyer, K.E., & Lynch, C. (2018). i think we should...: Analyzing elementary students' collaborative processes for giving and taking suggestions. https://doi.org/10.1145/3159450.3159507.

Tschan, F. (1995). Communication enhances small group performance if it conforms to task requirements: The concept of ideal communication cycles. *Basic and Applied Social Psychology*, *17* (3), 371–393. https://doi.org/10.1207/s15324834basp1703_6.

Waddock, S.A., & Bannister, B.D. (1991). Correlates of effectiveness and partner satisfaction in social partnerships. *Journal of Organizational Change Management*, *4*(2), 64–79. (Publisher: MCB UP Ltd) https://doi.org/10.1108/EUM0000000001192.

Zhang, J., Scardamalia, M., Lamon, M., Messina, R., & Reeve, R. (2007). Sociocognitive dynamics of knowledge building in the work of 9- and 10-yearolds. *Educational Technology Research and Development*, *55*(2), 117–145. https://doi.org/10.1007/s11423-006-9019-0.

Zhi, R., Price, T.W., Marwan, S., Milliken, A., Barnes, T., & Chi, M. (2019). Exploring the impact of worked examples in a novice programming environment. In: Proceedings of the 50th acm technical symposium on computer science education, pp. 98–104. https://doi.org/10.1145/3287324.3287385.

Zhong, B., Wang, Q., & Chen, J. (2016). The impact of social factors on pair programming in a primary school. *Computers in Human Behavior*, *64*, 423–431. https://doi.org/10.1016/j.chb.2016.07.017.

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

**Transparency on Re-use of Material** The authors confirm that this manuscript has not been published elsewhere and is not under consideration by another journal. This manuscript extends a work that has been previously published at Artificial Intelligence in Education 2021 in Lecture Notes in Computer Science, vol 12748 which can be found at https://doi.org/10.1007/978-3-030-78292-4_14. The original work reported on the hidden Markov model states and their transitions. This manuscript includes additional analyses of post-survey partner satisfaction scores and their relationship with the hidden Markov model as well as additional figures and insights about the general flow of a co-creative session.

Springer Nature or its licensor holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.

## Authors and Affiliations

**Amanda E. Griffith[1] · Gloria Ashiya Katuka[1] · Joseph B. Wiggins[1] · Kristy Elizabeth Boyer[1] · Jason Freeman[2] · Brian Magerko[2] · Tom McKlin[3]**

Gloria Ashiya Katuka
gkatuka@ufl.edu

Joseph B. Wiggins
jbwiggi3@ufl.edu

Kristy Elizabeth Boyer
keboyer@ufl.edu

Jason Freeman
jason.freeman@gatech.edu

Brian Magerko
magerko@gatech.edu

Tom McKlin
tom@thefindingsgroup.org

1 University of Florida, Gainesville 32607, FL, USA

2 Georgia Institute of Technology, Atlanta 30332, GA, USA

3 The Findings Group, Decatur 30030, GA, USA