New Orleans JAZZED about Engineering Education

Paper ID #16785

CONSIDER: A Novel Approach to Conflict-Driven Collaborative Learning in Engineering Courses

Mr. Swaroop Joshi, Ohio State University

Swaroop Joshi is a Ph.D. student in Computer Science and Engineering at The Ohio State University. His interests include Computer-Supported Collaborative Learning, Game-based learning, the use of technology in the classroom as well as a range of problems in software engineering.

Dr. Neelam Soundarajan, Ohio State University

Dr. Neelam Soundarajan is an Associate Professor in the Computer Science and Engineering Department at Ohio State University. His interests include software engineering as well as innovative approaches to engineering education. This will be a joint paper with my PhD student, Swaroop Joshi. He will be the first author, I will be the second author.

CONSIDER: A Novel Approach to Conflict-Driven Collaborative-Learning in Engineering Courses

Piaget's classic work on how children learn showed that when learners engage in critical discussions with peers who have ideas that conflict with their own, that contributes effectively to their developing deep understanding of the concepts involved. Building on this foundation, we have developed a novel and powerful approach to collaborative learning that exploits the power of on-line technologies to enable engineering –more generally, STEM– students to develop thorough understanding of technical topics through collaborative learning. Our approach, as we show, has a number of important advantages over most approaches to face-to-face collaborative learning. We have implemented a prototype web app, CONSIDER, based on our approach and used it in two Computer Science and Engineering courses: a graduate level theory of programming languages course, and an undergrad principles of programming languages course. It was very well received, with 15 out of 22 students in the grad course, and 13 out of 21 students in the undergrad course indicating, in a post-discussion survey, that the approach provided them a better opportunity to learn compared in-class discussions. We present a summary of the survey results, along with the theoretical underpinnings of the approach and some details of the prototype implementation. We also present our design for the next set of experiments with the CONSIDER tool.

1 Introduction

The type of conflicts of opinions and the ensuing argumentation seen in the broader public sphere would make it difficult for one to imagine that any type of conflict could ever be collaborative, let alone a driver of effective learning. But researchers in learning sciences have been studying what Andriessen [1] calls *collaborative argumentation*, which "can help students learn to think critically and independently about important issues and contested values". Indeed, Piaget [2], as part of his classic work on how children learn, shows that cognitive conflict arising from differences between different learners' understanding of important concepts, and the exploration and possible resolution of these differences by having the learners engage in critical discussion with each other, can be a powerful force in driving children's learning and in helping them develop deep understanding. In this paper, we report on a novel and powerful approach to collaborative learning that builds on the foundation of cognitive conflict and exploits the power of on-line technologies to enable graduate and undergraduate engineering students and, more generally, STEM students, to develop their understanding of technical topics through collaborative learning driven by cognitive conflict.

There are some challenges in developing such an approach. First, there are important differences between the children that Piaget's work addressed and college students, especially with respect to their willingness to participate in discussions with their peers. While most young children, after

perhaps an initial period of reluctance, willingly engage in such discussions, many college students, particularly in engineering courses, tend not to. This may especially be the case with female students and students from underrepresented groups[3]. Second, in many engineering courses, large class sizes (with 40 or more students) and short meeting times (around 55 minutes) are the norm. This makes it difficult to arrange students in groups of 3–4 each and have each group engage in deep discussions in class without each group disrupting other groups' discussions. Third, faculty, even those who may buy in, at a conceptual level, to the value of collaborative learning, tend to be reluctant about devoting class time to such activities since they are concerned about the likely negative impact on topic coverage. Our approach not only addresses these issues, it also has a number of other important advantages over in-person collaborative learning.

The paper is organized as follows. In Section 2, we review background theories related to learning driven by cognitive conflict; we also briefly review a number of approaches in the broader area of collaborative learning including those that use computing technology to help students engage in collaborative learning. We describe our approach in Section 3 and also briefly describe the prototype implementation of a web app based on the approach. We have used it in two computer science courses: a graduate level programming languages course in Spring-2015, and an undergraduate principles of programming languages course in the Autumn-2015 semester. In the grad course, a course assignment, in the form of a discussion on how Lisp, being a functional language, differs from imperative languages like C++ and Java, was conducted using our app. In the undergrad course, the activity was about the 'static' mechanism in languages like C++ and Java, and how it can be implemented in the interpreter project that was part of the course. After the completion of this activity, in each course, students were asked to complete a survey about their experiences in using the tool. In Section 4, we present an analysis of the survey results which suggest a very positive effect of the approach on students' learning, and highlights the importance of various features of our approach. We conclude in Section 5 with a brief summary and plans for future work.

2 Background

Our approach builds on two key notions that have been used successfully in various branches of learning sciences over the past few decades: Cognitive Conflict Driven Learning and Computer-Supported Collaborative Learning.

2.1 Cognitive Conflict Driven Learning

Piaget's original ideas, including those on cognitive conflict triggering learning in children were further elaborated and expanded by various learning scientists and applied to K–12 as well as college education. For instance, Doise and Mugny[4] conducted various studies about how cognitive conflict impacts learning. Their work showed that the other learner(s) who held the conflicting views did not need to be *physically present*, as long as the learners in question saw the conflicting

views as being those of *peers*. While triggering cognitive conflict *is possible* even without engaging with peers (e.g., via refutation text[5] instead), combining it with peer-interaction has major advantages. First, interaction with peers encourages the student to verbalize the cognitive conflict. Such verbalization of the conflict, and not the presence of conflict alone, is important for improving learning[6]. Second, the learner is forced to consider the alternative explanations offered by the peers and evaluate them on equal terms given that the cognitive conflict facing the learner is caused by the difference between the learner's own position and a *peer's* explanation rather than one offered by an authority figure such as a teacher or a text. In the latter case, a learner may simply accept the alternative explanation without critical evaluation. By contrast, when the (cognitive) disagreement is with the peers, the learner is forced to evaluate the alternatives critically since, for all she knows, it may be the *her own* position that is correct rather than the peer's.

Next we discuss some commonly used learning strategies in which peer discussions is an important part. Jigsaw[7] is a classroom technique where each student is placed in a *home* group and in an *expert* group. Each student in a home group is assigned a distinct topic. Students leave their home groups and join other students with the same assigned topic, forming the expert group on the topic. They explore their topic thoroughly and then return to their home groups; the student is then responsible for teaching his or her home group the particular topic.

In the Think-Pair-Share approach[8], the instructor poses a conceptual question and asks students to think individually about their responses. Then the students pair up with a neighbor and discuss each others' responses. Finally, the instructor calls on some students to share their answers. The pairing is not necessarily based on cognitive conflict.

Team-based-learning (TBL)[9] is a comprehensive learning system, unlike these two 'activities' that can be inserted in an otherwise *regular* classroom. In TBL, students are organized into teams of five or six each, and remain in teams throughout the course. The course is organized into units, each two to three weeks long. Before the start of a unit, students are assigned readings. On the first day of the topic, students complete, as individuals, a short test on the topic. Immediately after, they take the same test as teams, coming to consensus on answers; this step is effective only if there is a cognitive conflict among the students. The final step is a short lecture by the instructor focusing on common problems shared by many teams. The rest of the two to three week period is spent on activities that require the teams to apply the concepts and techniques to increasingly challenging problems.

Cognitive conflict plays a more central role is *peer-instruction* (PI)[10]. In PI, each student individually answers a conceptual multiple choice question, submitting the answer via a *clicker*; then the students turn to their neighbors and, in groups of 3 or 4, discuss the question; after a few minutes of discussion, each student again answers the same question. During the discussion time, the instructor walks around the room, observing the discussions but not participating in them. Mazur[10] reports that the percentage of students who, following discussion with their peers, change their answer from a wrong choice to the correct one far exceeds the percentage who change from the correct choice to a wrong one, demonstrating the power of collaborative learning driven by cognitive conflict. But there are a number of limitations with PI, mostly related to the fact that it is a

classroom technique:

- Since the multiple-choice question is about the topic being discussed in the lecture, students may not have had enough time to think about it deeply;
- The groups are formed primarily based on which students happen to be seated next to which other students, rather than on the basis of ensuring cognitive conflict in each group;
- Some students may dominate their groups irrespective of whether they have the right answers or not;
- The amount of time spent in the discussion is limited; hence, students who need time to formulate their arguments may not contribute effectively.

As we will see in the next section, our approach addresses all of these limitations.

2.2 Computer-Supported Collaborative Learning

Computer-Supported Collaborative Learning (CSCL) is a branch of the learning sciences that is "concerned with studying how people can learn together with the help of computers"[11].

CSILE (now KnowledgeForum) was one of the earliest CSCL systems[12]. A group of students using CSILE focuses on a specified relatively broad problem and begin to build a database of information about the topic. There is opportunity for reflection and peer review of each others' contributions by students. More recently, some authors used wikis to allow users to add, modify, or delete content using a standard browser, to create a site that thoroughly explores a topic. But, unfortunately, many of those studies have not produced as good results as expected. For instance, Cole[13] conducted an experiment in a course on information systems with 75 students; it was organized so that lectures were in alternate weeks, the other weeks being intended for students to discover new material and post to the class wiki. Fully one quarter of the questions on the final exam were to be from the material that students posted. The expectation was that students would post content, edit each other's posts, and engage in collaborative learning. Halfway through the course there had been no posts to the wiki! As another example, Leung and Chu[14] report results of the use of a wiki in a course on knowledge management. The class was organized into four groups of 4–5 students, each with a leader responsible for coordinating the group's work. Each group had to use a wiki to work on its project. In all of the groups, most of the contributions, in some cases up to 90% of the total, were made by the group leader alone. Judd, Kennedy, and Cropper[15] report similar findings from a large course on psychology.

An important reason could be the fact that there is little or no structure to the activities in these uses of wikis. On the other hand, in our approach, the activities are designed to trigger cognitive conflict leading to students engaging in effective collaborative learning.

Some researchers have suggested that technology, which is indeed the backbone of CSCL, should be exploited to realize some unique possibilities:

- The fact that CSCL environments can record the interactions in detail allows researchers to *zoom in* and see what exactly is going on during the collaborative interactions[16], making it a richer design environment for the researcher;
- Computational media, being configurable and adaptive, can make *new* interactions possible, instead of just replicating the face-to-face ones[11].

Our approach records the interactions in details, as well as creates new interactions via anonymous and asynchronous comments in a structured way (discussed in the next section). Indeed, it is these new types of interactions that are at the heart of the power of the approach.

It is worth noting that CSCL emerges from a wider research area: Computer Supported Cooperative Work (CSCW)[17]. Some recent works that are on the cusp of CSCW/L include the work by Greiffenhagen[18] which discusses how learning can change due to introduction of technology, using a study where students learn about *Macbeth* by producing their own storyboards in a CSCW software. Another example is the study by Martinez-Maldonado et al.[19] that analyzes the effects of deploying and visualizing the teacher's script for small group idea generation, using interactive tabletops, etc.

3 Approach

3.1 The CONSIDER Approach

Our approach, named CONSIDER (an acronym for CONflicting Student Ideas to be Discussed, Evaluated, and Resolved), works as follows. Following the standard class lectures on a given topic, the instructor will create a two-part assignment, call it A. The first part of A, call it I for *initial component*, will be a central, conceptual question that can, ideally, be presented as a multiple-choice question with distractors chosen to correspond to common misconceptions about the topic; the second part, P, is a more, in-depth question, that is an extension of I , possibly including a substantial problem-solving component. The instructor posts I on the CONSIDER web app and each student in the course is required to, individually, submit his/her answer to I within 24 hours. The system, possibly with help from the instructor, then organizes the students into groups of 4–5 each with each group containing students who choose different answers for I.

The students in each group will then engage in a series of rounds, $R1, R2, R3, \ldots$, of *discussion*, each round lasting 24 hours. The goal of the discussion is to help each student in the group arrive at an answer to the assignment. The goal is *not* for the group to arrive at a consensus answer. Instead, the goal is to have each student in the group arrive at his/her own answer to the question after careful consideration and analysis of the ideas of all the students in the group.

Suppose G is one of the groups and has four students, S1, S2, S3, S4. Note first that the students will not know the identities of the other students in G with the system simply referring to them as S1, S2, etc. When S1 logs in for, say, round $R3$, she will see the posts made by all four students

in $R2$. In her post for $R3$, S1 will have to indicate (by clicking a *green/red/blue* button on the app) whether she *agrees with, disagrees with, or is neutral/unclear about* the posts made by each of S1,..., S4 in $R2$ along with an explanation (especially if she disagrees); and also include her current approach to the problem. Note that S1 has to indicate, in $R3$, whether she agrees/disagrees with her *own* post from $R2$; the point is that, she may have found the $R2$ post from, say, S4 so convincing that she no longer agrees with what she said in $R2$! Indeed, this is *precisely* the point of peer discussion based on different conceptualizations of a problem. At the end of, say, $R5$ – this will be decided by the instructor and will vary with the topic– each student will be required to *individually*, submit his/her final answer to the assignment, along with a brief summary of the discussion in his/her group. S1's grade for the assignment will depend *only* on the correctness of her final answer and the quality of her summary; so she will not have to worry about losing points for switching from the wrong to the correct answer.

While the basic idea of cognitive conflict driving collaborative learning is based on earlier work, our approach, by careful use of the power of on-line systems, not only addresses the challenges to students learning listed earlier, but also offers a number of other important advantages. Anonymous posting lets students participate freely, mitigating any prejudices or biases some students may have about others. Asynchronous discussions allow time for students to carefully study their peers' arguments and formulate their own. The structure imposed on the discussion by the carefully defined concept of *rounds* with each student making exactly one post per round ensures that every student participates effectively to the discussion and helps contribute to the learning of each student in the group. None of these features is possible, or at least practical, without the use of technology.

We have implemented the approach as a scalable, platform-independent web app, using Google App Engine and Python, making it ubiquitous, accessible from any net-connected device of choice of the user.

3.2 End-User Testing

We used CONSIDER in two Computer Science and Engineering courses in last two semesters in a large public university in the mid-western United States. In the Spring'15 semester, it was used in a graduate level programming languages course, and in the Autumn'15 semester, in an undergraduate principles of programming languages course. The main goal of the grad course, like similar courses in other universities, is to study formal ways of defining syntax and semantics of programming languages. The main topics are attribute grammars; operational, axiomatic, and denotational semantics of languages. For the undergrad course, the topics are studying programming language constructs, and design and implementation issues for different language families; grammars and parse trees; interpretation versus compilation; data types, binding and scope rules; language constructs for control and data abstraction. In each course, one of the assignments was conducted as a CONSIDER discussion.

In the undergrad course, the assignment was about the "static" mechanism in languages such as C++ and Java. All 43 students in the course participated. The question shown in Fig. 1 was used

In answering this lead-in question, pick the one answer that you think is most correct and *complete*; and provide a brief justification of your choice.

The static mechanism, when used inside a Java class, is used for the following reason:

(a) The "static" keyword is used for only one purpose: to flag the main() function of the Java program so that the system will know that is where the execution should begin. The "static" mechanism is not used for anything else in Java (unlike in C++ which uses it for other purposes).

(b) In some sense, "static" is essentially equivalent to declaring something to be "public" so that a variable or method of the class that is flagged as static can be used anywhere in the program.

(c) Part of what (b) says is correct; when a variable or method of a class is flagged as "static", it is indeed *potentially* usable from anywhere in the program; but only *if* it is also flagged as "public". If it is flagged as "private", it is entirely useless since the rest of the program cannot use it.

(d) Part of what (c) says is correct but only part of it. If a class variable is flagged as "static", there is only one copy of that variable and that will be shared by all instances of that class rather than each instance having its own copy. The variable will be accessible anywhere in the program if it is flagged as "public"; but if it is flagged as "private", it will be only accessible by *static* methods of that class.

(e) Oh, (d) is so close but not quite right! A static, private variable of a class may be accessed by *any* method of the class, not just static methods.

(f) It is a useless mechanism. There are no situations in practice where we would need to use it. It should be removed from the language!

Figure 1: Lead-in question used in the undergraduate Programming Languages course (Au'15)

for forming groups of students with conflicting ideas. This being the first course in the conceptual ideas underlying programming languages for most of these students, there is a possibility of misconceptions about how these mechanisms function, as well as about their intended usage. The goal of the assignment was to help improve student understanding with respect to both of these aspects of the *static* mechanism. Based on the students' answers to the lead-in question, i.e., the initial component, ten groups of 4 each, and an eleventh group of 3 students were formed in such a way that each group had students with differing ideas about how to address the main problem. They were asked to discuss, over two rounds, strategies for implementing a *tokenizer*, which is a key component of implementing any programming language. In one of the projects the students had already completed earlier in the semester, they had implemented a tokenizer for a simple programming language named *Core* using the static mechanism. The question posed here was to come up with an approach to implementing the same tokenizer *without* without using the static mechanism (and without using a global table of Identifiers which, in this particular context, would amount to using the static mechanism). In almost every group, some students who started with a wrong notion about how to implement this feature ended up rectifying their approach, and even those who

Lisp is a *functional* programming language, not an *imperative* language. What this means for a Lisp user is (pick the one choice that you most agree with):

(a) It doesn't really mean anything since Lisp is as powerful as any other language. It is just a criticism used by people who don't like Lisp.

(b) It means that there are no assignment statements in Lisp. So algorithms that use assignments (which is pretty much *all* algorithms) cannot be implemented in Lisp; instead, you have to come up with completely different alternatives which is often not possible.

(c) It is true that there are no assignment statements in Lisp. What that means is that we should use the other features of Lisp such as function parameters, defining new functions, etc. to get the same effect as having assignment statements.

(d) It is not just assignment statements that are missing. It is everything else: sequential composition, if-then-else, loops, you name it! What a bogus language!

(e) We talked about it in class but I have no idea what it means!

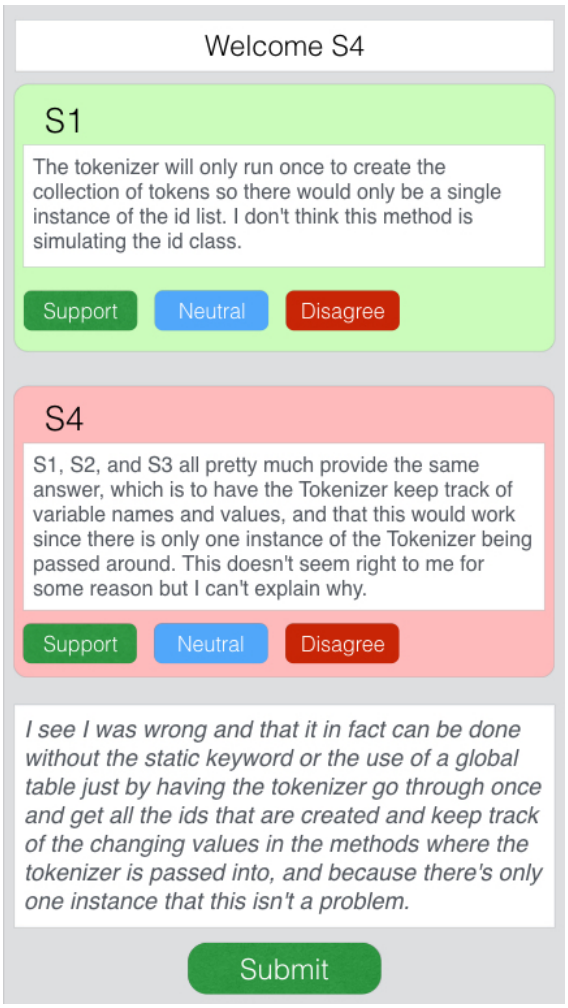
Figure 2: Lead-in question used in the graduate Programming Languages course (Sp'15)

started with the right idea were able to modify their strategy due to the CONSIDER discussions with their peers.

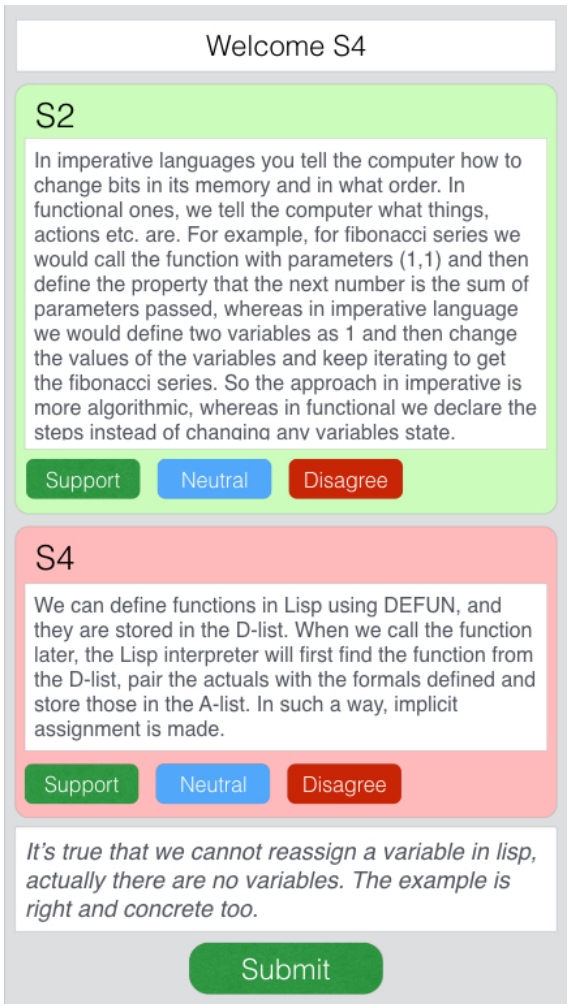
In the grad course, the assignment focused on how Lisp, being a *functional* language, differed from *imperative* languages, and how the effect of constructs of imperative languages, such as assignment statements, could be achieved in Lisp. Students in the course have typically not previously encountered functional languages, hence this tends to be a challenging topic for many of them. 39 of the 40 students participated in the activity; one student, because of a misunderstanding, missed the deadline for answering the lead-in question (initial component I of the assignment), and hence could not be included (he was given an equivalent assignment offline). The students were divided into nine groups of 4 each, and a tenth group having 3 students. The lead-in question used is shown in Fig. 2. The discussion that followed was about whether it was possible to implement a construct that was equivalent to an assignment statement in Lisp –and, if so, how.

Let's take a look at how the discussions look in the app. A snapshot of the discussion in one of the groups in the undergrad class is shown in Figure 3a. A student, whose alias is S4 in this group, had started with a position that the problem could not be solved within the given constraints (not using static mechanism or global table of Id objects). Other students in that group were of the opinion that it could be done even within the constraints. In $R2$, S4 read their $R1$-posts and became somewhat ambivalent with his position, which is shown in the middle box in the figure. S1, in his $R1$ post, shown in the top box in the figure, responded to S4 saying, "The tokenizer will only run once to create the collection of tokens so there would only be a single instance of the id list." S4 reads this post in $R3$, and corrects his position, as shown in italics in the box at the bottom in the figure. Posts of other students in the group are not shown here.

Figure 3b shows two students from the graduate class engaged in the discussion about implementing an assignment-like mechanism in Lisp. S4 of this group made an initial comment: "We can



(a) Undergrad course: discussion on use of static mechanism



(b) Grad course: discussion on implementing assignment-statement in Lisp

Figure 3: Sample discussions from the two courses

define functions in Lisp using DEFUN, and they are stored in the D-list. When we call the function later, the Lisp interpreter will first find the function from the D-list, pair the actuals with the formals defined and store those in the A-list. In such a way, implicit assignment is made." S2, on the other hand, had a slightly different take on the question. She explained that in imperative languages, bits are manipulated in memory which are assigned some meaning by the programmer, whereas in functional languages, the programmer tells the computer "what things, actions etc. are". She explained this with an example of Fibonacci numbers, and concluded, "So the approach in imperative is more algorithmic, whereas in functional we declare the steps instead of changing any variables' state." These were S4 and S2's posts, respectively, from $R1$. Upon entering $R2$, S4 sees S2's above comment and *agrees* with it by clicking the green button. Then she modifies her own position (shown in italics in the figure) to acknowledge that a variable could not be reassigned in lisp, *contrary to her initial claim*. They continued to discuss the details of how the functionality of

assignments could be implemented in Lisp.

4 Survey Data Analysis

In both classes, the discussion continued for three days, with most students logging in for 30–40 minutes each day, at their convenience, reading their peers' posts, formulating their responses to those, and posting those responses in the tool for that round. At the end, each student submitted her final answer to the problem each group discussed, and a summary of the discussion in her group. Feedback from the participants on their learning using the tool and various of its features was sought in the form of an anonymous on-line survey. 21 of the 43 undergrad students and 22 of the 39 grad students responded to the survey. All of them were Computer Science and Engineering (CSE) majors. Figure 4 shows the demographic details of the participants. Ethnicity information was not collected in the grad class.

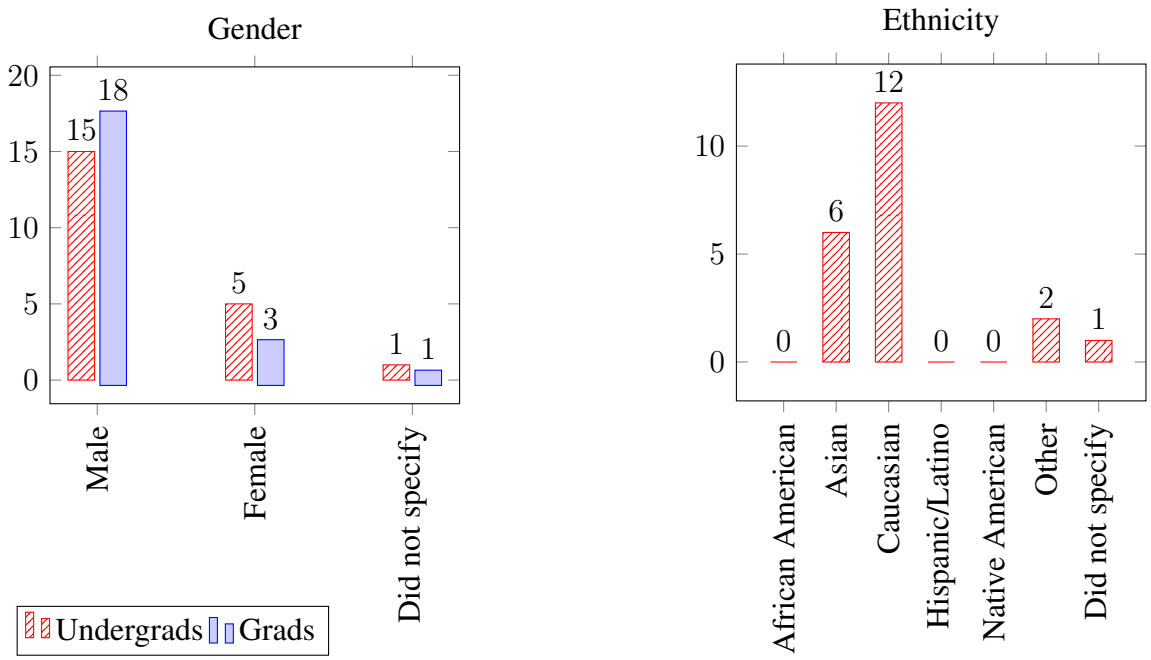


Figure 4: Demographic details

The responses to the question of whether the CONSIDER approach helped improve their understanding of the topic(s) highlight the benefits of our approach. For example, many students noted that they had not considered what might go wrong with their initial approach until someone in the group pointed it out; as a result, they were able to refine their answers. It is not possible for an instructor to point out such potential mistakes, let the student refine the answer, and again point out flaws, over and over again, for each of the 40-odd students in the class; nor can this happen efficiently unless students with conflicting ideas are grouped together. Another student commented: "Critiquing someone else's implementation helped in optimizing my own approach to solving the

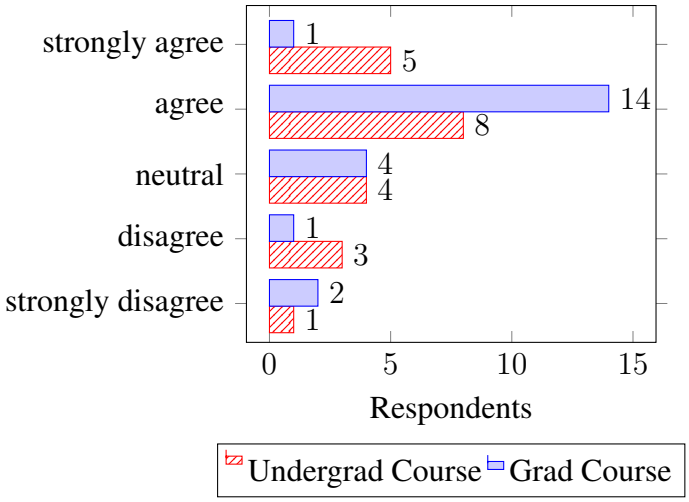


Figure 5: CONSIDER provides a better opportunity to learn compared to in-class discussions.

problem." Even a student who thought he had a good understanding of the topic even before the discussions started, conceded the usefulness of the approach, saying, "I had a good grasp on the topic from the start, but I was able to refine my answer –for this particular question– through the discussion."

About two-thirds of the students felt that the approach provided an opportunity to understand the topic in depth, compared to the teacher-led in-class discussions (by responding 'strongly agree' or 'agree' to the relevant question; see Figure 5). Some of them cited short meeting times as the reason for not being able to get into the details of a topic while discussing in-class. While some students felt that CONSIDER gave them space for a *personalized* discussion where they could contribute comfortably (presumably a reference to the fact that some students are not comfortable participating in the in-class discussions), some others felt that instructor's intervention would have been useful. One comment also mentioned that in-class discussions are "one time", referring to the short-lived nature of those, which is overcome by the on-line discussions recording all interactions.

The survey also asked students their feedback on the features of anonymity, rounds-based structure, and the duration and number of rounds used in the exercise. Pie charts in Figure 6 show student responses to each of those. We have combined the responses from both the classes together for these charts (For example, 11 out of 21 respondents in the undergrad class and 13 out of the 22 respondents in the grad class felt that the number of rounds was just about right. The pie chart reflects that the relevant field was selected by $11 + 13 = 24$ out of 43 respondents).

While a majority of respondents (56%) felt that the number of rounds (two discussion rounds and one summary round) was just about right for this discussion, in some groups the students converged in the first round itself, and the next round did not seem to add much value. One respondent mentioned that one of his peers was not clear enough in his comments, because of which he felt an additional round might have been useful. An interesting suggestion was that the tool should have the ability to get a vote from members of each group on whether they wanted another round or terminate the discussion following the current round; if the group voted unanimously to terminate

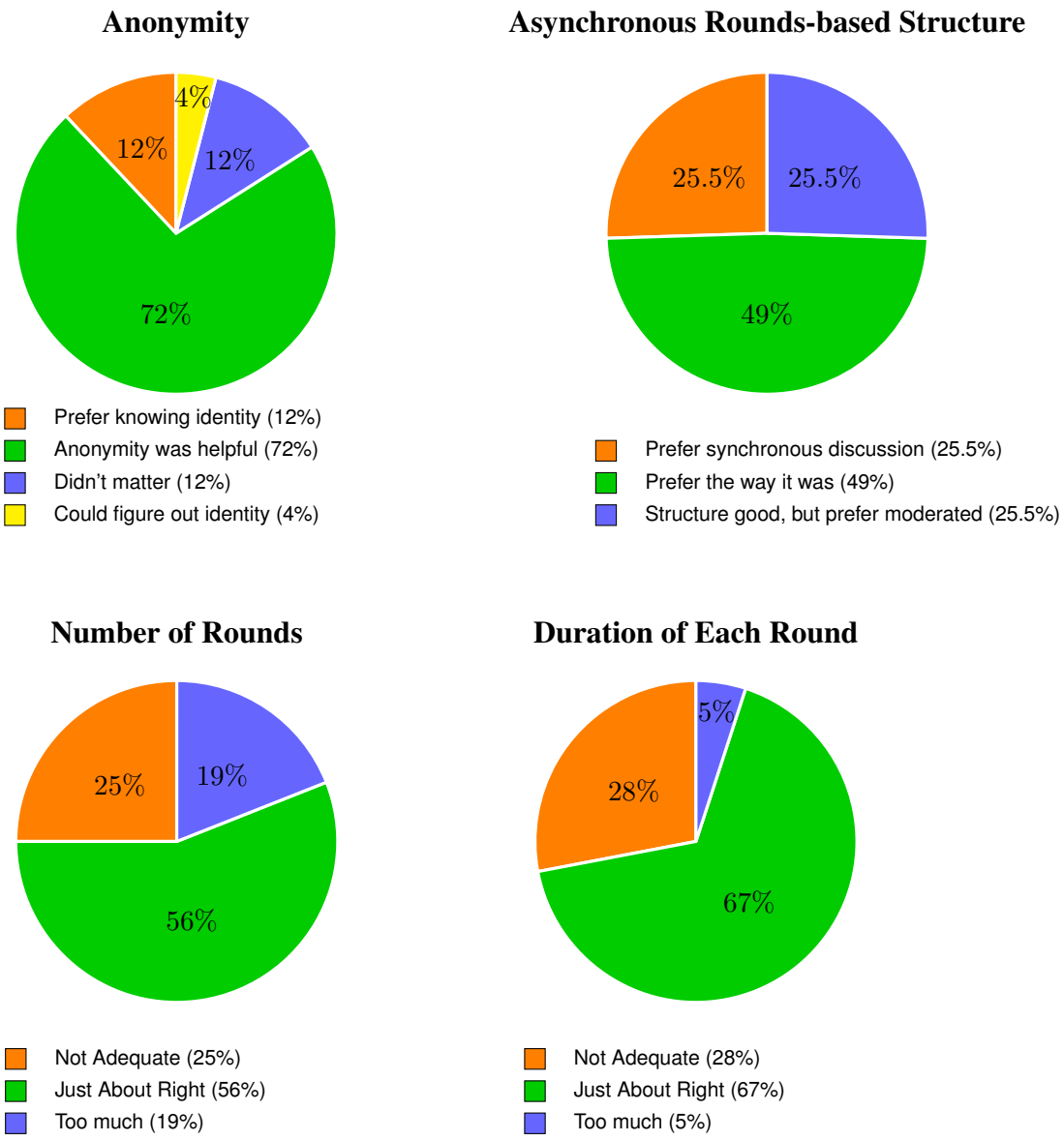


Figure 6: Survey responses on various features of CONSIDER

the discussion, the discussion would terminate, all students in the group would be so informed and each student in the group would have to submit his/her final answer and summary of the group's discussion. This would be a very useful feature, giving control of the discussion to the individual groups; and we are in the process of adding it to the next version of our app.

On the duration of each round, which was 24 hours, there were expected responses with 67% of the students saying it was just about right. Although a few students suggested having shorter rounds (12 hour), some other students commented that it should not be shorter since different people may have different schedules, and not everyone can find the same time of the day to log in and contribute to the discussion meaningfully. Some students also mentioned that 24 hours was good enough time

for them to reflect on others' comments and respond to it, without feeling rushed into it, as well as to not let their interest wane.

Two unique features of our approach are anonymity of the posters and asynchronous rounds-based structure of the discussion. The on-line survey asked for students' opinions on these features. 72% of the respondents felt that anonymity was helpful. Their comments on why they felt it was useful are in line with the reasons why we included it in the approach in the first place: it made sure that "answers were discussed without prejudice", and not having anonymity "may lead to (personal) conflicts at times." Some of the students who responded in favor of knowing the identity felt that way since it would allow them to continue discussing the topic in detail with the peers, even after the assignment is over. Another student's comment was along the same lines, but also hinted at a social advantage of disclosing identity, which we had not thought of. He commented that most students do not know each other, and some are shy of asking technical questions in class, but after engaging in discussions with a classmate for a few days, if they come to know who they are talking to, they may want to continue to collaborate with those even beyond this assignment or classroom. In future versions of CONSIDER, we may allow disclosing the identities of group members after the discussions, at the discretion of the instructor, possibly if *all* group members agree to it.

The asynchronous, rounds-based peer-discussion is another unique feature of CONSIDER. Opinions on this aspect were, somewhat to our surprise, rather mixed. About half of the respondents liked it the way it was, while about a quarter said they would prefer a synchronous discussion. Remaining respondents were ok with the structure, but preferred having the *instructor* intervene at some point in the discussion. Some of the respondents were okay with knowing the teacher's opinion *eventually*, which, in the form of an in-class session to clarify any remaining confusion on the topic after the exercise is over, is a practice we would recommend. But some other students felt that the instructor should provide feedback on *each* round. This would be essentially equivalent to grading 40-odd assignments *every day* for the duration of the activity, which is not practical. More importantly, it effectively means that if the group is unable to arrive at a consensus, it should be able to seek instructor intervention to settle the dispute "instead of", as one respondent put it, "arguing among themselves and wasting time." In addition to the practical problem with instructor's time mentioned above, we believe there is another, deeper problem reflected by such comments, related to what Rick and Guzdial[20] call a "lack of collaborative culture" in STEM education. These comments seem to reflect the widespread notion that discussing and arguing with peers about their conception of a technical topic does not (usually) contribute to learning. But this is contrary to our understanding of how learning happens, particularly in the case of collaborative learning, defined broadly as "a situation in which two or more people learn or attempt to learn something together"[16]. In their analysis of why their wiki-based collaborative learning environment was successful in English literature and architecture classes, but not in STEM courses, Rick and Guzdial identified one of the reasons to be "competitive nature of the courses and single answer questions". The same is reflected here with the student trying to get to the (presumably one, correct) solution of the problem via the instructor, instead of trying to discuss with peers other possible approaches to the solution, an exercise he considers a waste of time. Consciously employing collaborative learning techniques in all levels of STEM education might eventually mitigate such

issues, and would help see learners as well as educators the value of collaborative learning.

Nevertheless, there were students who appreciated the importance of collaborative learning driven by cognitive conflict, and of the emphasis on discussing the topics with peers. We would like to conclude the discussion on survey data by presenting a comment that reflects this understanding: "Since nobody is really expert like instructor, we have to provide strong evidence to convince others (as well as ourselves)." It is important to note here that, unlike the previous two comments which asked for the instructor to intervene in order to make sure they learn, this comment indicates that they learn better *because* the instructor is *not* present, resulting in them being more vigilant about the arguments they produce – this is *precisely* the point of collaborative learning driven by cognitive conflict.

5 Conclusion and Future Work

We have developed a novel, online approach, called CONSIDER, that combines the strengths of conflict-driven collaborative learning and Computer-Supported Collaborative Learning. The unique features of our approach and their benefits are summarized in Table 1. We have implemented it as a scalable, platform-independent web application using Google App Engine and Python. We used it in a grad-level programming languages course and an undergrad programming languages course in Computer Science and Engineering. Discussion activity in the form of homework assignments, about implementing assignment-statement like capabilities to the functional language Lisp was used in the grad course, while the use of the static mechanism in languages like Java/C++ was discussed in the undergrad course. 39 and 43 students participated in the grad and undergrad course, respectively. Preliminary analysis of the discussion data suggests that the approach was very helpful in improving the participants' learning about the concept. In almost all the groups, students changed or refined their solutions, based on comments of their peers, during the course of the discussion. In some groups, it was observed that, as a student critiqued another student's solution, he realized some caveats in his own solution and refined it. We continue to analyze the generated discussion data to understand how our approach helps students in learning Computer Science and Engineering concepts.

22 students in the grad class and 21 in the undergrad class completed the follow up survey for reflections on how the approach helped their understanding of the concept. All of the participants shared very positive feedback, highlighting the importance of the unique features of CONSIDER. In particular, students considered the feature of anonymously posting comments extremely helpful in participating freely and posting without prejudices. While almost $3/4^{th}$ (49+25.5%) students were supportive of the asynchronous, rounds-based structure of discussions, some of them (25.5% of the total) said they would prefer an intervention from the instructor, instead of discussing only with the peers. This opinion likely results from a lack of collaborative culture in STEM education, in which the value of collaborative learning, through discussing with peers, is not readily appreciated.

Feature	Resulting Benefits
Small group formation based on cognitive conflict	The discussion in each group is driven by the conceptual disagreement among its members; attempts to resolve it would lead to deeper understanding.
Anonymous posting in groups	(a) Students participate more freely; (b) The effectiveness of the discussion is not compromised by any gender/ethnic/other preconceptions some students may have.
Asynchronous, structured rounds-based discussions	Each student, whether quick on her feet, or prefers to think through subtle ramifications before posting, or anything in-between, participates equally effectively.
Online record of the discussion	(a) Students can come back and refer to their discussions and can continue to learn from the experience. (b) Instructors can look at the interactions and decide if further explanation is required for the topic.

Table 1: CONSIDER: Features and their Benefits

We plan to further evaluate the efficacy of the features of CONSIDER by designing careful experiments in coming semesters and using the tool in different engineering classrooms. We will compare CONSIDER with existing online discussion systems such as Piazza (`https://piazza.com`), which are quite popular in college courses, but lack the unique features of CONSIDER, like anonymity (it is optional in Piazza) and round-based asynchronous discussions (discussions in Piazza are threaded). This set of experiments will help us evaluate the effectiveness of these features of CONSIDER. Two independent topics of comparable difficulty will be discussed using Piazza and CONSIDER. An exam question on each topic will be asked in the finals. The students' performance on those questions will be statistically compared, in order to evaluate the effectiveness of CONSIDER on actual learning in comparison to existing tools like Piazza.

We are performing the study using the pattern of design-based-research[21]. This will have us engage in multiple, iterative formative assessments, and additional questions are likely to emerge as research questions are iteratively refined. This approach blends well with our software development pattern which is intended to follow an Agile process where features and functionality will be released and feature-by-feature testing will be done with end-users/learners.

Our tool is available as an open source software, which other educators can download and configure to use in their courses. It is highly customizable in terms of features such as number of rounds, duration of rounds, group size, etc., to suit their specific needs. It can be accessed at `http://go.osu.edu/consider`.

References

[1] Jerry Andriessen. Arguing to Learn. In R. Keith Sawyer, editor, *Handbook of the Learning Sci.*, pages 443–459. Cambridge University Press, 2006.

[2] Jean Piaget. *The early growth of logic in the child*. Routledge and Kegan-Paul Ltd., London, 1964.

[3] Sherry Hsi and Christopher Hoadley. Productive Discussion in Science: Gender Equity through Electronic Discourse. *J. of Science Education and Technology*, 6(1):pp. 23–36, 1997.

[4] Willem Doise and Gabriel Mugny. Sociocognitive conflict. In *The Social Development of the Intellect*, volume 10 of *International Series in Experimental Social Psychology*, pages 77–101. Pergamon, Amsterdam, 1984.

[5] Christine D. Tippett. Refutation Text in Science Education. *International Journal of Science and Math Education*, 8(6):951–970, February 2010.

[6] Patrick Jermann and Pierre Dillenbourg. Elaborating New Arguments Through a CSCL Script. In Jerry Andriessen, Michael Baker, and Dan Suthers, editors, *Arguing to Learn*, pages 205–226. Springer Netherlands, Dordrecht, 2003.

[7] Elliot Aronson. *The Jigsaw classroom*. Sage Publications, Beverly Hills, Calif., 1978.

[8] Jay McTighe and Frank T. Lyman Jr. Cueing Thinking in the Classroom: The Promise of Theory-Embedded Tools. *Educational Leadership*, 45(7):18, April 1988.

[9] Larry K. Michaelsen and Michael Sweet. The essential elements of team-based learning. *New Directions for Teaching and Learning*, 2008(116):7–27, December 2008.

[10] Eric Mazur. Peer instruction. In *AIP Conference Proceedings*, volume 399, pages 981–988. AIP Publishing, March 1997.

[11] Gerry Stahl, Timothy Koschmann, and Dan Suthers. Computer-supported collaborative learning: An historical perspective. In R.K. Sawyer, editor, *Cambridge handbook of the learning sci.*, pages 409–426. Cambridge, 2006.

[12] Marlene Scardamalia, Carl Bereiter, Robert S. McLean, Jonathan Swallow, and Earl Woodruff. Computer-Supported Intentional Learning Environments. *J. of Edu. Computing Research*, 5(1):51–68, February 1989.

[13] Melissa Cole. Using wiki technology to support student engagement. *Computers & Edu.*, 52 (1):141–146, January 2009.

[14] K. Leung and S. K. W. Chu. Using wikis for collaborative learning. In *Proceedings of the 2009 Intl. Conf. on Knowledge Management*, pages 1–13, 2009.

[15] Terry Judd, Gregor Kennedy, and Simon Cropper. Using Wikis for Collaborative Learning. *Australasian J. of Educational Technology*, 26(3):341–354, January 2010.

[16] Pierre Dillenbourg. What do you mean by collaborative learning? In *Collab.-learning: Cognitive and Computational Approaches*, pages 1–19. Pergamon, Amsterdam, 1999.

[17] Brian R. Webb. Opinion: Educational Research and Computer Supported Co-operative Learning. *Innovations in Education & Training Intl.*, 32(2):139–146, May 1995.

[18] Christian Greiffenhagen. Unpacking Tasks: The Fusion of New Technology with Instructional Work. *Computer Supported Cooperative Work (CSCW)*, 17(1):35–62, February 2008.

[19] Roberto Martinez-Maldonado, Andrew Clayphan, and Judy Kay. Deploying and Visualising Teacher's Scripts of Small Group Activities in a Multi-surface Classroom Ecology: a Study in-the-wild. *Computer Supported Cooperative Work (CSCW)*, 24(2-3):177–221, February 2015.

[20] Jochen Rick and Mark Guzdial. Situating CoWeb: a scholarship of application. *I. J. of Computer-Supported Collaborative Learning*, 1(1):89–115, March 2006.

[21] Sasha Barab and Kurt Squire. Introduction: Design-Based Research: Putting a Stake in the Ground. *The Journal of the Learning Sciences*, 13(1):pp. 1–14, 2004.