Active and Cooperative Learning

Introduction and Overview

In a traditional lecture format, the instructor prepares a lecture on the material and, at the appointed day and time, enters the class and gives a speech. The professor expects students to have read the material outside of class and perhaps have done some homework assignments. But inside the classroom, the professor is the “sage on the stage.”

Active learning (Bonwell91) involves students in the classroom by having them read, write, act, move, problem solve, discuss, etc. Involving students in the process gives them a deeper understanding of the material and leads to higher order thinking about class materials. The professor becomes the “guide on the side” in this model, creating an environment and activities that help students to learn. This document discusses how active learning can be done in the classroom as a substitute for pure lecture.

The basis for active learning is that people learn more and better when they are participants in the learning process. Teachers and professors are active learners when they prepare for class. That process can include reading the textbook, researching other reference sources, organizing the material into a logical sequence, and developing examples to illustrate the concepts. The fact that this process deepens our knowledge is reflected in the value that is placed on student presentations. Most people recognize that you learn more when you have to explain something to someone else.

By definition, Active learning in an educational setting requires that students be active in the classroom, rather than sitting and observing the professor. This can include doing problems on their own or within groups, participating in discussions, answering questions that require more than just facts, writing essays, or any other activity that is not just listening to a lecture. It has been found that student attention span starts to dramatically decrease about 12-15 minutes after the start of a lecture. [Stuart78] So, even just breaking up a long lecture with some intervening activity allows students to keep their attention up, and therefore learn more. These activities can be as simple as asking a few thought provoking questions or something more involved like group problem solving.

Active learning can be a discussion in the “Socratic method” between the professor and the students. But it can also move to another level, where students work with each other in small groups. Cooperative or peer learning ([Johnson91] and [Millis98]) gives students the opportunity to work together to solve problems or discuss concepts that leads to greater understanding for everyone in the group. Group activities also give students the opportunity to improve their social skills. People work in groups both large and small in professional environments. Success requires the ability to work with a wide variety of people. Cooperative learning creates situations where students begin to develop those abilities.

Traditionally, group work is a project done outside of the classroom, but here I’ll concentrate on activities that can be done in the classroom. Groups can be informal, for example, those students who happen to be sitting near each other, or formally assigned by the instructor. Groups can be short-term for one or two activities or long-term for one semester to a number of years.

Though someone might believe that these techniques can only be done with small classes, there are professors who use these ideas with large classes as well. [Weimer87] For example, Professor Gerald Feldman uses active learning in his large physics classes at The George Washington University. At the start of the semester, every student gets four different brightly colored cards with the letters ‘A’ through ‘D’ on them. Periodically during his lectures, he will ask a multiple-choice question that requires the students to think about a concept or solve a
Activity Types

There are a number of types of exercises that can be used in an active and cooperative learning setting. These exercises include textbook exercises, algorithm tracing, drama/role playing, demonstration software, polls, and think-pair-share. The bibliography gives references for more details on these and many other techniques. For now, I will discuss a few that would be most appropriate to use in computer science, and more specifically an analysis of algorithms classroom.

Textbook Exercises

Exercises from the textbook or exercises based on those in the textbook are an effective and simple way to involve students in the classroom. After an explanation of the material, students can be assigned a problem from the book to work in the classroom either on their own, or in a group. Groups can be an effective tool, if the groups are chosen by the instructor to be balanced in terms of student ability. When groups are balanced, group members will answer the simple questions that others in the group have, thus freeing the instructor to handle the deeper and tougher questions. If the instructor is walking around the room during group work, students are more willing to ask questions because they will only “admit” their lack of knowledge in front of their group. This is less threatening, since the group has already shown it can’t answer the question, and so all are interested in the answer.

This technique provides a good way to easily “test” student understanding of the material. Textbook exercises are traditionally assigned as homework, but when the student works these alone, the potential for frustration and discouragement is increased because there is no ready source available to answer questions. In some cases, the problem may be a simple one that leads to a significant loss of time. When these are done in the classroom, a student has peers and the instructor to help him or her through the difficult spots.

When I teach analysis of algorithms, I have a heavy reliance on textbook exercises, which is why there are so many in this book. For example, when I talk about bubble sort, I will explain the algorithm in a couple of sentences and then spend about 5 to 10 minutes talking about the analysis of bubble sort. I then stop talking and let my students work some exercises on bubble sort. I will typically begin with a simple exercise that requires them to trace the algorithm, so that they will have a clearer understanding of how it works. I then will ask them to work one of the problems that requires them to make a modification to the algorithm. Depending on the time I have allocated to this topic, I may also have them do one of the exercises to prove something about bubble sort.

Algorithm Tracing

It is typical for an instructor to display an algorithm and then proceed to trace through it while the class watches. An effective alternative is to give the students the algorithm and let them do
the tracing in small groups. By doing the work themselves, the students will have a greater appreciation of the algorithm and its detail and complexity. Furthermore, when the instructor is tracing the algorithm for a group, a student who doesn’t understand a step or gets distracted is not likely to ask the instructor to back up because it disrupts the entire class. But if this happens in a small group, the request will only be addressed to a few people and so is more likely to be asked.

An alternative would be to give each group a different set of input data for the algorithm and have them record their work on overhead transparencies. Then the groups could share their results with the entire class, which could result in a deeper discussion of the different ways the algorithm behaved. This could be done, for example, with the calculation of the fail links for the Knuth-Morris-Pratt algorithm or the slide and jump arrays in the Boyer-Moore algorithm.

**Role Playing**

Though role-playing is typically thought of as a tool of the humanities or social sciences, it can also be effectively used in the computer science classroom. A discussion of token passing in a networking class could begin by having students discuss a controversial topic, but requiring that students can only say one sentence at a time and only when they have a “talking stick.” Information hiding and encapsulation can be illustrated by putting data into a paper bag during algorithm tracing and only allowing students “playing” the member functions to see inside the bag. Recursion can be demonstrated by acting out the algorithms, pushing and popping students off of the system stack portrayed as a line they form at the front of the classroom. ([Fleury97])

**Demonstration Software**

Software that will animate an algorithm can be found on the World Wide Web for a number of different algorithm types. Further, some of the programming exercises given in chapter 3 ask students to develop visualizations for various sorting algorithms. These visualizations can be very effective in helping students understand the different ways that sorting algorithms accomplish their task.

The Association for Computing Machinery’s Special Interest Group on Computer Science Education (ACM SIGCSE) supports a project that catalogs a set of education related links. This web site includes links for demonstration software and algorithm animation software. Renée McCauley maintains this site. It can be found at: www.cs.cofc.edu/~mccauley/edlinks/.

**Polls**

Polls can be a quick way to get class feedback on an answer a student or students have given before the instructor gives his or her views. If I ask the class for an answer to a question and I get two or more different answers, instead of correcting errors or identify the right answer, I will ask students to vote, by a show of hands, for the answer that they think is correct. I always include an “I don’t know” option as well. At this point, I can now further the discussion by asking students to give the reasons why they are supporting the answer they raised their hand for, or to explain why they disagree with one of the other answers. This gets the students to think more carefully and critically about their answers.

This technique is effective in increasing the level of participation in a class. Students are more willing to risk giving an answer because they know that they won’t be put on the spot and immediately told that they are wrong. This also gives other students the opportunity to correct errors, which is less threatening than having the professor correct them.

**Think-Pair-Share**

In the think-pair-share model, a question is posed to the class and the students are asked to write their answer to the question down on a piece of paper. In this first stage students work
individually. This provides an important opportunity for the students who may be Introverts on the Myers-Briggs Type Indicator ([Briggs76] and [BriggsMyers85]) to think about their answer before being asked to talk about it. The second stage asks students to pair up with a person near them. These pairs discuss what they have written and come to a consensus answer. If the question requires the construction of a list, perhaps the pairs are asked to pick the two (or any other appropriate small value that requires choices) most important or best of their individual items. The last stage is to call on pairs and have them share their short list with the rest of the class.

This process is very effective at getting a wide range of student participants. The think stage lets the Introverts compose an answer before the Extroverts shout out theirs. The pair stage allows students to gain an advocate for their answers (the one they are paired with) so that if they are challenged they have someone to defend them.

For example, this technique could be used when discussing graph traversal algorithms. Students could work on their own to determine the order the nodes would be visited, and then compare their answers with the student sitting next to them, before sharing with the class.

Group Formation

Groups are classified as either formal or informal based on how they are formed. Formal groups are chosen by the instructor according to some scheme so as to create heterogenous groups. Typically, this is based on ability and creates groups with a combination of average and above average students. Though groups should be as diverse as possible, it is best to not isolate one woman or one member of a racial minority in a group. Informal groups tend to be for quick activities and are frequently based on where students are sitting that day.

There is a preference for instructor chosen groups, which separates students who are good friends. When one group contains friends there is a greater potential for problems. For example, if one person is not carrying their weight in group work, a friend is not likely to raise that as an issue for fear of losing a friendship. Further, friends can form alliances based on external considerations and not the quality of the work or the needs of the group. Friends are also more likely to get “off task” with discussion unrelated to the work. This leads to problems in the group process, which has an impact on the performance of all of the group members.

Group work should also have some weight within the grade distribution. It has been reported that group work should be at least 20% of the final grade, and should include a peer evaluation component. ([Feichtner85]) The final grade component for group work should not be too high, or it will substitute for individual responsibility and accountability. Group assignments should have clear objectives and should not be doable by one person. Group work should also promote positive interaction between group members and help them develop social skills for a professional environment.

Active and Cooperative Learning in Analysis of Algorithms

Active and cooperative learning can have a significant impact on student learning especially in an analysis of algorithms course. I feel that there is no absolute right or wrong way to use these techniques as long as a few principles are followed.

• Know what you want your students to learn through the exercise. You should be able to give a clear and succinct answer to the question “What do I want my students to learn from this activity?” Exercises shouldn’t try to “teach” too many different concepts, because if they do, it is easy for some things to be lost. In this book, I’ve included exercises that are focused on one learning objective whether that be to understand how an algorithm works, to make a
modification to an algorithm so it behaves differently, or to prove something about what an
algorithm does or produces.

• Give your students enough time to do the work. It takes a group of people longer to get to a
solution than it takes one person. In a group, everyone must have the opportunity to
contribute, ask questions, understand the proposed solution, and come to a final agreement.
When you estimate how long you expect your students to complete a task, give them at least
twice that time to actually do it. You can always have an extra question handy in case they
finish early. You probably won’t need the extra questions very often. In this book, there are
a mixture of short and long exercises. This gives you the flexibility of adding an extra short
task if you need to keep a group busy while the others finish up the last question.

• Stay aware of what the groups are doing. By circulating around the room, you can observe
what the groups are doing and how they are doing it. This also places you nearby, making it
easier for students to ask questions. You can also watch the group process to make sure that
all students are given the opportunity to participate, and that disagreements are settled in a
professional manner.

The rest of this section will describe how I use this book to teach analysis of algorithms.
This information is presented as one model of how this book might be used, not as the only way
it could be used.

At the start of every semester, I get a list of my students along with their grade point
averages (GPA). I use this information to create groups of 4-5 students each. These may seem
like big groups, but all of the group work they will do is during class time, so scheduling is not a
problem. (For courses where the group work is outside of class, I use smaller groups.)

Outside of class, I begin by order the list of students by decreasing GPA. I calculate how
many groups I need by dividing the number of students by 5 and then rounding up. If this means
I need, for example, 6 groups, I go down the list, and number the students 1, 2, 3, 4, 5, 6, 6, 5, 4,
3, 2, 1, 1, 2, 3, 4, 5, 6, etc. Assigning students to groups in this way, produces groups that are
pretty balanced. If you instead numbered the students 1, 2, 3, 4, 5, 6, 1, 2, 3, 4, 5, 6, etc, all of
the students in group 1 would have GPAs higher than their comparative student in groups 2, 3, 4,
5, and 6. In other words, the first student in group 1 would be higher than the first student in all
the other groups, and the second student of group 1 would be higher then the second student of
the other groups and so on. This makes group 1 stronger than group 2, which is stronger than
group 3 etc.

Groups in my courses are fixed for the entire semester. Some of the studies I’ve read
indicate that it can take as much as one-half of the semester for a group to gel. If the groups
change during the semester, the students will spend time working through group issues all
semester, which distracts from course content. I tell students that in a professional environment
they will either need to work with others on their “team,” or transfer to another part of the
company or another company. Learning to work with others is easier than transferring, and
college is a good time to learn those skills.

Because of the drop/add period at the start of the semester, I will sometimes hold off a
few classes before deciding on the final groups. This typically depends on the level of the
course. There is much more movement of students in and out of a CS 1 class than there is in an
upper-level required course. So, when I teach CS 1, I spend the drop/add period doing activities
that get the (typically) freshmen majors to know each other and that also demonstrate the value
of active and cooperative learning. (See [Silberman96].) For upper-level classes with a more
stable enrollment, I give the group assignments as early as the first class.

On the first day of class, when I am describing the structure of the course, I also talk at
length about active and cooperative learning. I mention that studies show that active and
cooporative learning increases student learning and material retention. I also mention that
students are more comfortable with the material and what they know at exam time. I talk about both their responsibilities and my responsibilities for these techniques to work. I let them know that even though I’m not lecturing, there is a lot of work answering student questions and grading student work. I make this part of the presentation very strong and upbeat, because I want the students to believe, as I do, in the effectiveness of active and cooperative learning. If they believe in the value of active and cooperative learning, they will do the class preparation work needed for them to succeed. This, by the way, is the same amount of work we normally want our students do. I find that by explaining why I’m using active and cooperative learning my students are more involved and are more likely to actually prepare for class.

The size of a textbook is a critical component if students are going to do reading assignments. This book has about 250 pages of material. When you allocate time for exams and course administration, students will be expected to read 20-25 pages per week. This is not an overwhelming amount and so students are more likely to do the reading. A book that covers the same material in 500 pages would require students to read 40-50 pages per week. Only the most studious of students will be likely to do that much reading for one course. Can, however, this material be adequately presented in just 250 pages? I feel it can because the exploration of the topics is not done by reading someone else’s exposition, rather the material is learned by having the students do the exploration themselves. This cornerstone of active learning is what leads to deeper understanding and longer retention of the course material. This is done in the book with the exercises and study suggestions.

I explain what the students can expect each class right at the start of the semester, because my students need to understand the process if it’s going to work. I typically teach my courses as two 75 minute periods each week of the semester. I start every class meeting with a short mini-lecture on that day’s material. This topic may be one that is complex and difficult to understand, or that is critical for my students to fully understand for the rest of the course or their careers. I try to keep this mini-lecture to 15 – 20 minutes, which means I can’t explain everything in the reading. I then assign one of the exercises from that section of the book for the students to work on with their assigned groups. When a group finishes the first exercise, I assign a second exercise on either that material or the next topic depending on what I want to accomplish. Studies of intellectual development indicate that people advance when they first get reinforcement for what they know and then get challenged to reach a bit farther. Therefore, in each of the sections of this analysis book, I have tried to put simple exercises that require a trace of the algorithm as well as more complex ones. I’ve also tried to put in multiple exercises of each type so that some could be assigned in class and others as homework. The study suggestions at the start of the chapters give another set of data that the students can use to trace the algorithms with the results given in Appendix C.

Some classes have just one mini-lecture that reviews all of the topics for that day and the rest of the class is group exercises. In other classes, where I want the students to work with the first topic before discussing the next, I will interrupt the group work to give another mini-lecture before moving them onto exercises for the next topic.

During the time students are working in their groups, I feel it is critical to walk around the room and check on my students. I’ve told my students that it’s my role to be available for questions about the material, but to also watch that their groups are working in a professional manner. I find that if I “join” their groups, they feel intimidated and stop talking or will turn to me to ask if they are correct. So instead, I stand near the groups, looking away from them, but eavesdrop on what they are doing. If it appears that I’m looking out the window or am reading a notice posted on the wall, students won’t realize that I’m really listening to them, and their group work goes on uninterrupted. If I see they are having trouble and need help, I will then “step into their circle” but will still wait for them to ask me a question before offering help or advice.
Many times they will find the solution on their own, which empowers them to do their own learning.

Regular evaluation is important to me. It is a way for me and my students to know how they are doing. At the end of class, every group turns in one exercise answer for the group. Exercises are graded and returned the next class along with an answer key so I don't need to review exercises in class unless they were really bad. I have found that it is easier to grade when I use a small point scale. My exercises are graded on a 5-point scale, but most of the time I usually give only 5, 4.5, 4, or 3 points. Since I know the students were there and worked, I figure it would be unfair to give less. So if an answer is correct, it gets a 5. If the answer is good but has small mistakes, it gets a 4.5. An “average” answer gets a 4. An answer that doesn’t have enough work or has significant problems gets a 3.

I do not take attendance. I require that each student sign his/her group's answer before it is turned in. When I grade the answers, I immediately see who was there for that activity, and I can record grades appropriately. Students who miss a class do not share their group's grade for that class, but instead receive a 0 for that activity.

I give class exams about 2-3 times a semester. The class exams have both an individual and group part that are done in class. This means that the individual part should take them about 30 minutes and the group part about 45 minutes. The longer time allocation for the group part is to give them time for discussion. The group exam has one or two harder questions from the individual part, and usually one additional question that needs group cooperation for completion. For example, if the individual exam has questions that require the students to (1) trace the output of an algorithm, and (2) write a recursive algorithm and create the recurrence relation for it, the group exam could repeat the second question and also have the group solve the recurrence relation.

Throughout the semester, I make clear that the active and cooperative learning components of my courses are designed for students to master the material. I give clear direction as to the content of the exams. I use last year’s exams as practice exams, so students know the general content and level of difficulty. I feel that my job is to give the students every opportunity to demonstrate what they have learned, not show them how much more than them that I know. I make clear from the start that the final exam is cumulative and all individual work, which gives students one last chance to show their level of mastery of the course material.

Peer Evaluation

Students need to know that they will have the opportunity to evaluate how their group functioned. Peer evaluations give them the opportunity to report on how each member of the group contributed. For the group process and the peer evaluation to be taken seriously, it must have an impact, even a small one, on the course grade. The evaluation can be an informal one, where students are given the task of writing a short narrative on their fellow group members, or it can be a more structured evaluation form. In my courses, I use a form that asks students to rate themselves and their group members on issues such as their preparedness, contribution to the group, willingness to let other participate, and how well they listened to that participation. The form asks students to rate how often the group members exhibit various positive behaviors. It gives a scale from 0 to 5 with the values labeled with names like Never, Rarely, Occasionally, Frequently.
Grade Record Keeping

Students always want to know how they are doing, and giving them feedback is a great encouragement. I use a computer spreadsheet to keep track of my grades. I have set up an area in the spreadsheet where group grades are entered. I then use the lookup function to have those grades "distributed" to the area for the individual students. I have a macro that will automatically print out the information for one student at a time. This is done by hiding the information for all the other students and then printing the spreadsheet. This process is fully automated so that I can, with one command, print a sheet for each student showing the scores I have on record for them. I usually distribute these to the students after each exam. Each student gets a sheet that shows their information only (for confidentiality) and also has information on average scores, the high and low scores. Though the spreadsheet also has the student's rank in the class, that column is minimized for printing. I do this because students told me that they get discouraged if they have a low ranking even if the entire class is doing well.

A fictitious sample of the spreadsheet and macro are available for download at the web address below. Details on how the spreadsheet and macro work can also be found there.

Final Thoughts

There are a few hints that I would like to share from my experience with active and cooperative learning. There are those who claim that active and cooperative learning is more work than pure lecture. That might be the case if you are in a field where there is not much change and lecture notes can be used year after year. I find, in a rapidly changing field like computer science, that I need to prepare my courses fresh each time. So, for me, the move to active and cooperative learning was not necessarily more work, just different work. In the past, my time was spent preparing 3 lecture hours per week. I now prepare class exercises and less than 1 lecture hour per week, with the rest of the time going to grading student exercises.

As you might imagine, there is a lot of paper that gets shuffled. This can consume a great deal of class time especially when you have exercises and answer keys that are returned each class. At the start of the semester, I create two folders for each group. Each one has the group number on the front and the group members’ names on the inside. One of these folders is used to return group work and the other is used to return individual work. For example, instead of having to pass out 40 pieces of paper to 40 individuals, I would give out 8 packets to 8 groups of 5 students and the groups then distribute the papers. I give back exams individually, but use the second set of folders to sort them. In this way, I only approach each group once instead of running around the room as I return the exams.

With active and cooperative learning, there are a lot of things that I'm taking to class. This includes the textbook, handouts, student work, transparencies, and announcements. At times, I may also have some “toys” I’m going to use for demonstrations. I use a separate canvas bag for each of my courses in a semester. As I’m preparing things for class, I just put them into the bag, as they are ready. When its time for class, I just grab the bag and I'm off.

I have prepared a web site that has information on active and cooperative learning and links to other active and cooperative learning sites, but also includes samples of my peer evaluation form and my grade keeping spreadsheet and macro. This web site is at:

www-cs.canisius.edu/~mcconnel/active_learning.html
Bibliography


