Active learning and its use in Computer Science

Jeffrey J. McConnell
Canisius College
Computer Science Department
Buffalo, NY 14208
mcconnell@canisius.edu

Abstract

Student learning and the depth of the student's knowledge increase when active learning methods are employed in the classroom. Active learning strategies are discussed in general computer science course work and as used in a theory of computation course. Difficulties with active learning and techniques for dealing with these are also presented.

1. Introduction

Active learning [1] gets students involved in activity in the classroom rather than passively listening to a lecture. This activity can be reading, writing, discussing, solving a problem, or responding to questions that require more than factual answers. The idea is to get students thinking about the material. This is important, because students who are passive have a decline in concentration after 10-15 minutes in a 50 minute lecture [4].

Further, the act of learning is not passive. As faculty, we learn actively. In preparing lecture notes, we read, compare what we have read with our experiences, synthesize the information into coherent notes, and develop examples that illustrate the concept. This leads to greater understanding of the material. Unfortunately, we then use this understanding to lecture to our students depriving them of this journey of discovery. By carefully involving the students on this path to knowledge, we can increase student depth of understanding of the material, increase student comfort with the material, and improve student confidence. In most sciences, the value of active learning is already realized and implemented through laboratories, or in computer science, through programming projects. The ideas present here are to expand this to include activities in the classroom that replace part of the lecture.

If active learning is so successful, why is it not used more frequently? This is because there is a perception that active learning has higher risks. There is fear that content will have to be taken out to put active learning in, that pre-class preparation time is higher, and that active learning is not appropriate for large classes. Perhaps the largest fear is giving up control of the classroom -- a lecture lets the professor decide what to say when, where student centered activities may raise questions that the professor was not planning to discuss.

These fears are real, but surmountable. To cover the content, give students the responsibility for learning the factual material so that they can apply it in the classroom discussion. I have had students write on course evaluations that they didn't need to read the book before class because my lecture would tell them what they needed to know from the chapter. If students believe that they will hear a lecture similar to the text material, they will not be willing to read the text.

For faculty that re-use class notes year after year, developing active learning strategies will take more time than pulling notes out of a filing cabinet. In a field as rapidly changing as computer science, notes need to be done frequently enough that this should not be as great of a concern. Further, as you develop ideas for active learning, you will find that they can be applied across a number of different courses.

Active learning strategies allow you to control the level of risk. By selecting short, highly structured and well-planned activities, the level of risk is fairly low. Involving students by asking a series of questions about the current topic allows the teacher to control the direction and content of the discussion but still makes students active. Breaking the students into small groups, and letting them independently solve a problem is a much higher risk but can prove to be highly rewarding.

2 Active learning techniques

As mentioned above, strategies for active learning are equally valid in many computer science courses. This section will describe generic techniques for active learning in computer science. The next section will show the application of some of these techniques to a theory of computation course.

2.1 Modified lecture

As was mentioned, student attention begins to decline after 10-15 minutes of lecture. Further, we have all been in lectures where something catches our attention, causing us to 'miss' part or all of the next point. A strategy to handle both of these is to lecture for 10 minutes and then take a 5 minute 'break.' During the break, students discuss their notes with the person next to them filling in gaps and correcting misunderstandings. Alternatively, an activity that leads to a discussion would be to pose a question and then employ the 'think-pair-share' technique. In this technique, a question is posed to the students who then individually write an answer within a one to two minute time limit. Each student then 'pairs' up with the person next to him/her and they discuss their answers, possibly developing a new one. The instructor can then start a discussion or the next lecture topic by asking a few pairs to 'share' their answer with the class.

In a computer science class, you could pose questions like:

- What will happen if I change this input?

- What could happen if I don't include this conditional check?

- Why is it important to reduce the number of operations in sorting algorithms?

- Why are all four conditions necessary for deadlock to occur?

If the professor has a computer in the classroom, and the questions relate to the structure of an algorithm or its behavior the answers can be tested in real time. Even better, if the students have computers at hand, groups can get together and try solutions on their own. In the later case, the instructor must be careful that the students think about their answers first, or the activity will degrade into a hacking session.

2.2 Algorithm tracing

Instead of tracing the execution of an algorithm in a lecture, break the students into groups and have them trace the algorithm. For example, to compare sorting algorithms break up the class into groups of four students each. Assign one student as the algorithm tracer, one to keep track of the variable values, another to record the number of additions/multiplications performed, and the last to record the changes to the list. By providing each team with transparencies and markers, teams can easily display their answers to the rest of the class. By then running an implementation of the algorithm, group results can be quickly compared with the actual answer

2.3 Demonstration software

In a classroom with a projection unit connected to a computer system running demonstration software, the professor has a powerful tool to have students interact with the ideas of computer science. By dividing the students into groups, you can ask them to predict what will happen to the output based on changes to the input or the algorithm.

This set up also allows students to formulate 'what if' questions as they are trying to understand an idea. For example, students trying to understand the nature of recursive algorithms can see the effect of input on the results.

A large amount of software of this type has been developed for a number of areas of computer science. This is readily available through the internet.

2.4 Physical activities

Role playing can be a powerful tool in the computer science classroom. To have the student understand network protocols, you could have the class passing a ball around during a discussion, letting students talk only if they have the ball. Students experience first hand the operation of Token Passing Protocols. Similar activities could be used to simulate the flow of information through circuits (students are chips passing information back and forth), or the execution of parallel algorithms (groups do different parts simultaneously).

I have recently used an activity of this type to illustrate the concepts of objects and classes in an introductory programming class. In this class, I used paper bags to represent the objects, with the private data inside the bags. Students in the class were given code fragments representing class methods as well as the main program that used these classes. We then executed the code, with constructor functions getting new bags for the objects they create,

and with the bags (objects) moving around the room, being opened only by class methods. This clearly illustrated to the students the actions of class methods and the relationship between those methods and the private data.

3 Theory of computation experiences

I have been teaching theory of computation for a number of years at both the upper undergraduate level and in the sophomore year. Our students have taken theory of computation in the first semester of their sophomore year, after having completed a one semester computer science overview course and CS1. They take CS2 and a one-credit hour C programming course concurrently with theory of computation. The purpose of the theory course is to introduce students to finite state machines, regular expressions, grammars, Turing Machines and the limits of computing. At the sophomore level, the course is more practically oriented than proof based.

In the past three offerings of the course, I have used three different teaching styles. For Fall 1992, the course was taught with a typical lecture style of presentation. Students did regular homework assignments, took three semester exams, and a final exam.

For Fall 1993, the course had a different look. Students were assigned into groups of three that were static for the entire semester. (For information on using group work see [2].) Class time was broken into two 10 minute mini-lectures, each followed by a 15 minute group problem solving exercise in the classroom. One group would then present their solution to the class. There were regular homework assignments done as a group. These exercises and homeworks were taken directly from the problems in the text. Students still took three semester exams individually, but then the groups also reworked the same exam over the weekend and turned in a group solution. So the students took each exam twice, once alone and once with their group.

For Fall 1994, the course became a combination of the two previous offerings. Because of my fear that the groups were not working effectively enough to learn the material, the classroom became closer to a lecture hall. I did keep some active learning by asking questions that lead the students in the solving of problems. However, the students no longer did group exercises in the classroom, though they continued to work on homework and exams in groups outside of class.

3.1 The results

When I decided to implement active learning in my class, I wanted some way of checking to see if there was any benefit to my students. The active learning literature (for example [3]) said that there would be a positive effect, but I wanted to test it. There are numerous studies with large student populations that show positive influence on students' grades when active learning is employed. Even though my classes were small, I wanted to test to see if I would find a similar improvement.

Since I do not distribute copies of my final exams, and since they are cumulative over the entire semester, I thought that could be a mechanism to test how my students had done under these various techniques. So, for each of the three semesters mentioned above, I used the exact same final exam. I also collected information on the grade point averages (GPA) for the college level course work that my students had completed before starting this course. There were a few students that had just transferred to my institution or

53

were just cross-registered from other institutions. They have been excluded from this study because there is no reliable grade point average information for them.

To avoid differences in grading schemes that are very likely for exams graded over a three year period, all exams were duplicated and regraded. Exams for the three classes were mixed together before regrading to help eliminate any unconscious biasing of the results. Further, I developed a five point scale for each question that was used to take out the differences in point levels between the different questions. The general meaning of these point values is given in table 1.

Points	Description
4	Completely correct
3	Generally correct with a few errors
2	Shows correct technique but it is poorly applied
1	Generally incorrect but it shows a small understanding of the problem
0	Completely incorrect

Table 1 Meaning of point values for regrading

After excluding transfer students that had no established GPA, the sample of students involved in this study was small (Fall 1992 - 11, Fall 1993 - 11, Fall 1994 - 16). The results of an ANOVA single factor analysis between class and GPA, and class and exam score showed that there was no significant difference in the GPAs of the students in the three classes, but that there was significant difference in the exam scores (table 2). Students in the classes where some (Fall 1994) or many (Fall 1993) active learning techniques were used performed better on the final exam.

	Fall 1992	Fall 1993	Fall 1994	F ratio
GPA	2.69	2.62	2.68	0.034
Exam Score	8.00	13.82	12.88	3.440**

Table 2. ANOVA single factor analysis
Average GPA and Average Exam Score

In order to assess the strength of the effects between the variables, a correlation matrix was calculated. Since the exam scores in the second year were higher than the third, the classes were renumbered for correlation with the other variables (Fall 1992 = 1, Fall 1993 = 3, Fall 1994 = 2). The correlation between GPA and final exam score was 0.64 (p<0.1). The correlation between GPA and class (-0.04) indicates independence between GPA and class assignment. The correlation between class and exam score was 0.37 (p<0.05). Thus the correlation between class and exam score was not confounded by the assignment of students with different GPAs to different classes.

In summary, the student sample in the three offerings of this course did not significantly differ in their ability, base on their entering GPA. This is important, because, as is expected, the results show that students with higher GPAs will perform better on exams. Since each class had similar GPA distributions, there was no difference in student ability that could have skewed these results. The important point, however, is that students in the courses with active learning did better than those that were not. Further, the addition of significant group activities in the classroom (Fall 1993 vs. Fall 1994) seemed to help further, but to a much less noticeable amount.

4 Conclusion

Based on my experiences, I now appreciate the need to have a balance between instructor presentations, and student activities. It seems that some lecture is necessary to discuss those things that the instructor views as critical or difficult to understand. It is also important, however, that there be some activity to get the students active in the classroom.

Adopting active learning techniques can be risky for faculty, but the risk can be minimized by choosing short well structured activities. As your comfort level with active learning increases, riskier strategies can be tried. Though the loss of control can be scary at first, I have found myself invigorated and look forward to the challenge of the active learning.

The reality of today's higher education in the United States is that students do not seem to be as interested in learning as they once were. By employing active learning strategies, students not only learn content, but process as well. This makes them better students in later courses, and better professionals after finishing their degree.

References

1   Bonwell, C. and Eison, J. Active Learning: Creating Excitement in the Classroom., *ASHE ERIC Higher Education Report No. 1,* Washington, D.C.: The George Washington University, School of Education & Human Development, 1991.

2   Feichtner, S. and Davis, E. Why Some Groups Fail: a survey of students' experiences with learning groups. *Organizational Behavioral Teaching Review,* Vol. 9, 1985, pp. 58-73.

3   McKeachie, W., Pintrich, P., Lin, Y., & Smith, D. Teaching and learning in the college classroom: a review of the literature. The University of Michigan, Ann Arbor, Michigan, 1987.

4   Stuart, J. and Rutherford, R. Medical Student Concentration During Lectures. *The Lancet,* Vol. 2, September 1978, pp. 514-516.