Active Learning in Computer Science (2012)

Abstract—**This Active learning in Computer Science paper describes on how student learning and the depth of the student's knowledge increase when active learning methods are employed. Active learning strategies are discussed in general computer science related and as used in a theory of computation course. Some of the difficulties with active learning and techniques for dealing with computer science are also presented.**

Table of Contents

I. Introduction

THIS report is based on Active Learning in Computer Science. Active learning 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. Moreover, the act of learning is not passive. For faculty, they learn actively. In preparing lecture notes, they 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. 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.

Active learning is very successful, but it is 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. 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. 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.

II. 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 Research that supports active learning.

A. 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:

If the professor has a computer in the classroom, and the questions relate to the structure of an algorithm or its behaviour 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 latter case, the instructor must be careful that the students think about their answers first, or the activity will degrade into a hacking session.

B. 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

C. 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 setup 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.

D. Physical Activities

Role playing can be a powerful tool in the computer science classroom. To have the student understand network protocols, the instructor could have the class passing a ball around during a discussion, letting students talk only if they have the ball. Students experience firsthand 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. Math

III. Research Supporting Active Learning

One study has shown evidence to support active learning Bonwell and Eison (1991) state that active learning strategies are comparable to lectures for achieving content mastery, but superior to lectures for developing thinking and writing skills.

According to another study by Armstrong (1983), students who receive a formal education learn better when they are actively engaged in the learning process as opposed to those who do not partake in the learning process. In addition to that, Armstrong (2012) provided some examples of active tasks as writing papers, problem-based projects, and experiential exercises (e.g., role-playing)

IV. Benefits of Active Learning

The Active Learning process is a good way to develop students' analytical skills. Activities such as role-play exercises allow students to examine a question or topic from different angles. Students learn to appreciate different points of view and to build a strong case in support of the view they are presenting in class.

Students not only read the required material for a tutorial - they are stimulated to reflect on it too. Group activities allow them to explore their ideas in a small and supportive environment and to develop a reasoned argument. They are given the opportunity to listen so that they can critically evaluate the arguments and work of others as well as their own.

By allocating specific tasks and activities to be prepared for lectures or tutorials, students develop their independent learning skills. It provides an environment where they can apply the theoretical knowledge gained from a lecture directly to their own work.

V. Conclusions

Active learning has many faces. There is no simple connection between the organisation of classroom activity and students' learning. Even if the students are formally engaged in goal setting and decision making, or are engaging in small group’s discussion, there is no guarantee that they will learn anything. Conversely, effective learning can take place in whole-class instruction where the teacher keeps center stage. It seems crucial to take into consideration the degree to which students decide to participate. They might not make that decision unless the teacher is able to communicate with them.

One way to secure the students motivation is to involve both the cognitive and the affective sides. If the students have an emotional connection to the subject, engagement is more likely.

VI. References

[1] https://marketing.wharton.upenn.edu/files/?whdmsaction=public:main.file&fileID=3459

[2] http://news.wustl.edu/news/Pages/13517.aspx

[3] http://en.wikipedia.org/wiki/Active_learning_(machine_learning)

[4] http://en.wikipedia.org/wiki/Active_learning