INCREASING UNDERGRADUATE INVOLVEMENT IN COMPUTER SCIENCE RESEARCH

Collberg, C., Debray, S. , Kobourov, S., and Westbrook, S.

Department of Computer Science, The University of Arizona, Tucson 85721, Arizona, U.S.A.

ABSTRACT

Current undergraduate Computer Science curricula are generally built around a set of traditional lecture-oriented courses where the student is a passive recipient of knowledge. While easy to implement, such a model has the drawback of presenting the field as a static corpus of facts and techniques. It does little to challenge and engage the brightest of students, or prepare them to participate directly and actively in a highly dynamic and rapidly evolving field. Nor does it give them a sense of engagement, belonging, and ownership in this body of knowledge. This paper describes our experiences with addressing this situation via a model that aims to get undergraduates exposed to, interested in, and involved with research early in their academic careers. We use a set of closely related research-oriented courses, starting with research seminars suitable for freshmen and sophomores, and leading up to advanced projects for juniors and seniors. These courses have the effect of engaging talented undergraduates in research early in their college careers. T This approach has led to a dramatic increase in the amount of undergraduate involvement in academic Computer Science research in our department in the last few years, and resulted in numerous research publications and awards.

1. INTRODUCTION

An effective way to improve the undergraduate learning experience is to involve students in research. Research experiences enable students to apply what they learn in traditional classes, challenge them to learn new things, involve them with research teams, encourage them to go on to graduate school, and may entice them into research careers. Undergraduate research experiences have proved especially useful as a means to encouraging women and other underrepresented groups to pursue careers in computing (Cuny & Aspray 2002).

The impact of an undergraduate's research experience, both for the student and the faculty supervisor, increases directly with the length of research involvement. This is due both to the (cultural and technical) learning curves involved, which take an investment of time and energy from both the student and the supervising faculty member; and to the fact that a student who engages in research over a longer time can contribute more to, and benefit more from, the research project. For these reasons, the benefits of undergraduate research, for both the students and the supervising faculty members, are greatest when students become engaged in research early in their student careers.

However, a pragmatic problem arises here: a typical college freshman or sophomore has very little idea of what Computer Science research involves, whether or not it is interesting, which faculty members are interested in engaging undergraduate researchers, and how to go about finding a suitable research project to work on. Unfortunately, by the time undergraduates figure out the answers to these questions, they generally do not have enough time left in their college careers to allow for a significant length of involvement in research. Another problem is that faculty members are often reluctant to involve students who cannot yet contribute fully to a research project. The problem is especially acute for women and underrepresented minorities, who have fewer role models they can turn to for mentoring and guidance in these matters.

The situation is, in many ways, qualitatively different from that in other sciences, where motivated students can get acquainted with research via simple and routine laboratory activities, e.g., preparing or maintaining lab specimens or equipment. Such activities provide an opportunity for students to learn the basic tradecraft of a field while observing, and interacting with, more advanced students and researchers, and thereby learning the nuts and bolts of how research is carried out. In Computer Science, the analogous routine maintenance tasks would be activities such as installing or updating software, writing simple scripts, running benchmarks, collating data, etc. Our experience has been that, unlike the physical and biological sciences, where students can get an early introduction to laboratory research via simple yet useful activities, it is much more difficult for undergraduate students to become involved in Computer Science research

early in their careers. For example, tasks such as installing or upgrading software very often requires special administrative privileges that are unlikely to be given to a first- or second-year student. Other tasks, such as running benchmarks or collating data, are in many cases so easily automated (e.g., via scripts) that it is often quicker and simpler for a faculty member or graduate student to go ahead and do the work rather than spend the time to explain how it should be done—and explain the intricacies of the relevant software systems—to a young and inexperienced student.

The underlying problem here is one of mutual fear of risk. From the perspective of a faculty member directing a research project, early undergraduates represent many unknowns in terms of their aptitudes and suitability for research, making it a risk to invest time, research funding, or other resources on them; many faculty feel that the risk is not worth taking, and simply refuse to consider undergraduates—no matter how bright—for their research. From the perspective of the early undergraduate, the idea of research represents many unknown, both in terms of general issues such as what it involves and, what the benefits might be, as well as the specifics of working with a particular faculty member on a particular research project; many undergraduates are therefore leery of investing time and effort on getting involved with research, and often prefer to simply focus on their coursework. For some undergraduates—even very bright ones—there may also be an intimidation factor at work: students who have only seen the end product of research, but never participated in creating them, may have the (mistaken) impression that elegant solutions to complex problems are born "fully formed" from the researcher's mind, and therefore consider themselves inadequate to the task.

The resulting lack of undergraduate research involvement shortchanges both sets of participants. On the one hand, faculty who do not consider undergraduates for research can miss out on some extremely bright and highly motivated students who can make excellent research contributions (we have been fortunate to work with many such students). And, on the other hand, undergraduates who shy away from research can miss out on a host of positive experiences that not only reinforce and augment their classroom lessons, but also provide great emotional rewards as well as credentials that can be valuable for getting jobs or admission to high-quality graduate programs. This problem is especially acute for second- or third-tier educational and research institutions in developed countries as well as institutions in developing nations, because very often the best and brightest of the undergraduates from such institutions will move to other institutions, for reasons of either opportunity or prestige, for their post-graduate studies.

This paper describes a model we have developed to reduce these perceived risks, to both faculty and students, via a process that is relatively low-cost for both sets of people, but which can succeed in overcoming some of the barriers we have found hinders the participation of Computer Science undergraduates in academic research. Our experiences indicate that such an approach can be useful for increasing undergraduate involvement in research.

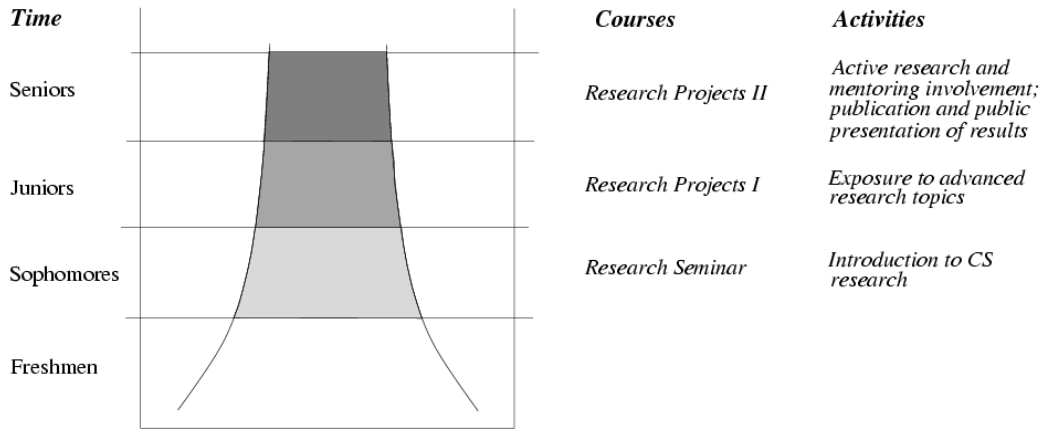

Figure 1: A pipeline model for undergraduate research

2. A PIPELINE MODEL

As noted above, a mutual lack of information between research faculty and undergraduates poses a barrier to the early involvement of undergraduates in academic research: researchers do not know the interests and strengths of undergraduates, while undergraduates have no exposure to the nature of academic research and therefore do not know

what it involves and whether it is worth participating in. This problem is greatest for students early in their academic career, i.e., first- and second-year students, whose research involvement is likely to be of the greatest benefit for both the student and the faculty member. By the time this lack of information has been resolved to some extent—for example, for a fourth-year student who has taken many classes, acquired greater technical expertise, and has gotten to know the research faculty and vice versa—the benefits of research involvement are reduced, simply because the length of time the student is able to be involved with a project is relatively not very long.

To address these problems and create an environment where undergraduates can be systematically exposed to the research process and given opportunities to become involved with research projects, we have developed a pipeline model for undergraduate research (see Figure 1). The idea is to begin by exposing students to ongoing research projects and activities as early as their sophomore year; then have them gain experience with more advanced topics and research issues in project-oriented class sections in their junior year; and, finally, integrate them into active research groups, and have them carrying out their own research activities by the time they are seniors.

While this model is designed to allow a student to incrementally become more involved in research, we also want to include students, such as transfer students, who might not have entered the pipeline at the beginning and those who become interested in research later in their academic experience. Although proposed seminars and projects are targeted at specific classes of students, they are not restricted to students only at those levels. Whereas we would like students to go through the full stages of the pipeline, some students might participate in only one or two of the proposed courses before they graduate. Thus, an even greater number of undergraduates will have the opportunity to gain research experience.

Our proposed model also provides a lot of opportunities for students to mentor other students at earlier stages of the pipeline. For example, in our projects, we typically have PhD-level graduate students mentoring senior-level undergraduate researchers, who in turn mentor junior-level students, and so on. Such mentoring activities are both educational and satisfying for the individuals involved; they are also helpful in cultivating a sense of community among the students. Such affinity research groups have been shown to be a successful way of retaining and recruiting students (Alvarado and Gates 1997, Alvarado *et al.* 1999).

2.1. Components of the Pipeline Model

In this section we describe the various components of our pipeline model for integrating the research experience into undergraduate education.

2.1.1. Research Seminar

The first component of the pipeline is an informal research seminar, aimed at first- and second-year Honors students, that has two goals. One goal is to acquaint the undergraduate students with the research faculty and some of the technical details of ongoing research projects. The other is to give students some anecdotal exposure to the process of doing research—how research goals are formulated, problems identified, and solutions developed, evaluated, and refined—as well as more subjective aspects of research, such as the frustration of dead ends and the satisfaction of formulating elegant solutions to tricky problems.

This seminar is structured as a weekly meeting, roughly an hour in duration, where a faculty researcher (and, occasionally, post-graduate or even more advanced undergraduate students) discusses his or her research. The topics presented vary widely, and range from strictly technical presentations pertaining to a research project, at one extreme, to very personal discussions of one's own perspective on the meaning and experience of doing research, at the other. We attempt to keep the depth of technical sophistication at these presentations to a level that allows the typical undergraduate to grasp at least the essential ideas, if not the full technical details, of the problem(s) being addressed, why they are interesting, the challenges that have to be addressed, and the broad outlines of the solution and how it was obtained.

Since on average each research faculty member is asked to make a single such presentation per semester, the faculty workload induced by this seminar is an hour per semester, i.e., two hours per year. This is small enough that it does not

pose a significant imposition upon researchers. The additional time commitment for students is one hour per week, which is also not excessive.

2.1.2. Research Projects I

Once students have an idea of the different research projects under way, their goals, and the kinds of problems they address, many (though not all) of them decide that they want to participate in research. The next step is to give these students some experience with actually doing some research on their own. The expectation here is that these there will be more of a time and resource investment from both the faculty and the students; however, it is important to ensure that, even if after this experience a student decides not to pursue undergraduate research at greater depth, there is not a large loss of time, effort, or resources to either the faculty member directing the research, nor to the student (otherwise, the faculty member and/or the student would very likely be reluctant to get involved). To this end, we make these research projects be part of undergraduate "Honors section" projects for courses such as *Systems Programming*, *Compiler Design*, *Algorithms*, *Theory of Computation*, and *Operating Systems*. In addition to the regular assignments and exams, honors students select one of several projects and work in teams to solve a well-defined problem. As an example, a group of students in a third-year *Systems Programming* class built a distributed software system for playing checkers over a local area network—a project that required them to learn about, implement, and integrate issues such as low-level Unix sockets programming (for communication over a network) and graphics and user interfaces (for interacting with users).

The structure of these projects is to have a weekly meeting, moderated by a faculty member, where the various groups give progress reports and discuss technical issues relating to their projects. Additionally, students put in roughly three to five hours of time per person per week. The aim is to set sights that are ambitious enough to make for a meaningful research experience for the students; nontrivial enough to be of interest to the faculty member directing the projects; and still keeping the additional workload manageable enough, for both the students and faculty members involved, to ensure that the demands of these projects does not become too overwhelming.

2.1.3. Research Projects II

After participating in Research Projects I, the students have some experience of the process of doing research, and know better whether it is something they enjoy and want to pursue further. At the same time, the faculty members involved have come to know the students better, have a better understanding of their strengths and weaknesses, and are in a position to decide whether they wish to invest time and resources on deeper and more challenging research projects with any of them. This sets the stage for Research Projects II, the final stage of our pipeline, where students become involved with research groups either individually, or in small groups of two or at most three people.

We have found that projects suitable for undergraduate research share a number of characteristics. The following is a partial list of such characteristics.

1. *A project should not require overly sophisticated technical background to get started.*

   There are two problems with requiring a great deal of technical expertise for students to get started in research. First, it delays their initial research involvement, and thereby limits the amount of research accomplishment that is likely. Second, it serves to exclude a larger number of students because of the particular classes they have (or have not) taken.

2. *The eventual research goal should be clearly defined and should be attainable via a well-defined series of small steps.*

   This is important for two reasons. First, for students just starting out on research, it is important to establish a series of attainable targets, both to make steady progress and to bolster their confidence. Second, it allows

students to understand whether or not they are making progress, and if not, identify the sticking points (which can, in turn, suggest how they might be addressed).

3. *There should be groups of related projects. However, individual projects should be sufficiently independent.*

   Having groups of related projects helps establish communities of people who can share ideas and help each other with problems. It also helps with the mentoring process discussed in Section [*]. However, individual projects should not be too tightly dependent on one another. Otherwise, if for some reason a particular project does not make progress at the expected rate, or is abandoned, it can cause problems for other research projects, potentially delaying or disrupting them significantly.

Our undergraduate research projects were selected to meet these desiderata as far as possible. They formed part of our regular research programs, in the areas of algorithms, graph drawing, programming languages and compilers, and software security. This had the benefit of allowing the undergraduate researchers to be supported with our research grants, which allowed them to devote more time to research. It also provided greater mentoring opportunities to the more senior student researchers because several students were working on related research problems.

2.2. Strengths of the Pipeline Model

The pipeline model and the affinity research group model together offer a number of strengths. It is a complete package that integrates research into undergraduate education at every level of the pipeline, starting with freshmen and sophomores all the way up to seniors.

For students just starting their college careers, involvement in a research seminar engages them in a very different way than regular classes: problem-solving requires a very different skill set that is often not exercised in regular courses. We believe that such a loosely-structured and cooperative setting (rather than the regimented and competitive setting of typical undergraduate courses) is more likely to successfully engage students from underrepresented groups.

The research problems driving the junior-level project-oriented sections typically come from the ongoing faculty research projects, e.g., code obfuscation, software watermarking, graph drawing and visualization, code optimization, etc. This gives students a sense of ownership and make the work more real. An additional benefit is that most of our projects have a significant combination of theoretical and practical aspects.

Incorporating these aspects into the undergraduate curriculum and exposing students to their interplay has the benefit of illustrating both (1) how theory is applied to solve practical problems; and (2) how practical problems give rise to new theory. Overall, our approach offers a broad framework: the model can be replicated at other institutions as well, possibly with different concrete research projects and seminar courses depending on the research foci of the faculty.

3. EVALUATION

While a precise quantification of the effects of our approach to involving undergraduates in academic research is beyond the scope of this paper, we have some anecdotal evidence of its efficacy.

The first, and most direct, effect of increasing undergraduate research involvement is in the number of research publications coauthored by undergraduate student researchers. This is illustrated in Figure 2. Prior to the start of this project, there was essentially no undergraduate involvement in research in our department; as expected, peer-reviewed research publications co-authored by undergraduates were also rare. After the initiation of this approach, however, the number of such publications has increased dramatically, going from none at all in 1999 and 2000, to a single paper in 2001, to seven in 2002 and 12 in 2003. While we cannot expect this rate of growth to be sustained much longer—for a fixed number of faculty available to supervise research, there will be natural limits on the number of students each faculty member has time to supervise—this suggests that our approach has been effective in significantly increasing a commonly used measure of research productivity.

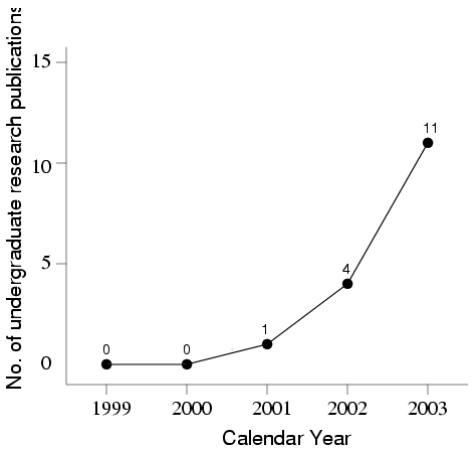

Figure 2: No. of peer-reviewed research publications co-authored by undergraduates

Also telling is the visibility of students involved with our research at the Department, University, and National levels. Prior to the initiation of this project, the "Computer Science Department Outstanding Senior" award, given to the student deemed the best graduating undergraduate student, was in most cases decided almost entirely based on the student's classroom grades; because of a lack of additional accomplishments beyond their grades, these students typically did not have much visibility at the university or national level. However, this changed as undergraduates became more involved with research. Since 2002, undergraduate researchers working with our research groups have won three "Department of Computer Science Outstanding Senior" awards, as well as a number of awards at the national level, in particular, a Runner-up and two Honorable Mentions in the Computing Research Association (CRA) Outstanding Undergraduate competition (CRA is an umbrella group of top academic, industrial, and governmental organizations involved with computing research in North America).

4. CONCLUSIONS

Undergraduates are generally not incorporated into academic research programs in Computer Science. This has the problem that academic researchers can often fail to utilize the talents of bright and talented undergraduates; and undergraduates may miss interesting and useful learning opportunities. Among the reasons for this problem are that undergraduate Computer Science students are very often not exposed to the research process until much too late in their academic careers, if at all; and that faculty researchers are unaware of the strengths and interests of individual undergraduates until late in the undergraduates' academic careers. This paper describes a model that allows undergraduates to be exposed to the research process, and get involved with it, in a systematic way. Our experience with using this model is that it has led to a significant increase in the amount of undergraduate research involvement in our department, with numerous concomitant benefits, such as publications at high-quality peer-reviewed conferences and journals that have been co-authored by undergraduates, increases in undergraduate Honors theses, and increased visibility and recognition of our undergraduates at the university and national levels.

REFERENCES

Alvarado L. and Gates A. 1997. Affinity research groups: Attracting and retaining women in computing. *Proceedings of the Tenth International Conference on Women in Higher Education*, pages 10—15.

Cuny J. and Aspray W. 2002. Recruitment and retention of women graduate students in computer science and engineering: Results of a workshop organized by the Computing Research Association. *ACM SIGCSE Bulletin* 34(2): 168—174.

Brass, P. *et al*., 2003. On simultaneous graph embedding. *Proceedings of the* 8th *Workshop on Algorithms and Data Structures*, pages 243-255.

Chandrasekharan, S. 2003. An evaluation of the efficacy of control flow obfuscation techniques against profiling and intelligent static attacks. Undergraduate Honors Thesis, University of Arizona.

Collberg, C., et al. 2002. AlgoVista: A tool to enhance algorithm design and understanding. Proceedings of the 7th Annual Symposium on Innovation and Technology in Comp. Sci. Education (ITiCSE).

Collberg, C., *et al*., 2003a. Error-correcting graphs for software watermarking. Proceedings of the 29th Workshop on Graph Theoretic Concepts in Computer Science, pages 156-167.

Collberg, C., *et al*. 2003b. SPlaT: A system for self-plagiarism detection. Proceedings of the International Conference on WWW/Internet.

Collberg, C., *et al*. 2003c. A system for graph-based visualization of the evolution of software. Proceedings of the ACM Symp. on Software Visualization (SoftVis), pages 77-86.

Collberg, C., *et al.* 2003d. Error-correcting graphs. Proceedings of the Workshop on Graphs in Computer Science (WG'2003).

Collberg, C., *et al.* 2003e. Tetratetris: An application of multi-user touch-based human-computer interaction. Proceedings of the 9th International Conference on Human-Computer Interaction (INTERACT), pages 81-88.

Erten, C., *et al*. 2004.. Exploring the computing literature using temporal graph visualization. Proceedings of the Conference on Visualization and Data Analysis (VDA).

Erten, C., *et al*. 2003. GraphAEL: Graph animations with evolving layouts. Proceedings of the 11th Symposium on Graph Drawing (GD).

Erten, C., *et al.* 2003b. Simultaneous graph drawing: Layout algorithms and visualization schemes. Proceedings of the 11th Symposium on Graph Drawing (GD).

Gates A., *et al.* 1999. Expanding participation in undergraduate research using the affinity group model. *Journal of Engineering Education* 88(4):409—414.

Heidepriem, Z. 2003. Detecting differences in java bytecode. Undergraduate Honors Thesis, University of Arizona.

Linn, C. and Debray, S. K. 2003. Obfuscation of executable code to improve resistance to static disassembly. Proceedings of the 10th. ACM Conference on Computer and Communications Security (CCS 2003), pages 290-299.

Sahoo, T. and Collberg, C. 2004. Software watermarking in the frequency domain: Implementation, analysis, and attacks. Proceedings of the ACM Symposium on Applied Computing (SAC 2004).

Schwarz, B. 2002. Post-link-time optimization on the Intel IA-32 architecture. Undergraduate Honors Thesis. University of Arizona.

Schwarz, B., Debray, S. K. and Andrews, G. R. 2001. Plto: A link-time optimizer for the Intel IA-32 architecture. Proceedings of the 2001 Workshop on Binary Translation (WBT-2001).

Schwarz, B., Debray, S.K., and Andrews, G.R. 2002. Disassembly of executable code revisited. Proceedings of the IEEE 2002 Working Conference on Reverse Engineering (WCRE).

Shaked, T. 2002. Value profiling in PLTO. Undergraduate Honors Thesis, University of Arizona.

Snavely, N. 2003. Optimizing and reverse engineering Itanium executables. Undergraduate Honors Thesis. University of Arizona.

Snavely, N.. Debray, S.K., and Andrews, G.R. 2002. Predicate analysis and if-conversion in an Itanium link-time optimizer. Proceedings of the Workshop on Explicitly Parallel Instruction Set (EPIC) Architectures and Compilation Techniques (EPIC-2).

Snavely, N., Debray, S.K., and Andrews, G.R. 2003a. Unscheduling, unpredication, unspeculation: Reverse engineering Itanium executables. Proceedings of the IEEE Working Conferece on Reverse Engineering (WCRE-03), pages 4-13. (Extended version to appear in IEEE Transactions of Software Engineering, 2005).

Snavely, N.,. Debray, S.K., and Andrews, G.R. 2003b. Unspeculation. Proceedings of the. 18th IEEE International Conference on Automated Software Engineering (ASE-2003), pages 205-214.

Yusufov, R. 2001. GUIDE: A system for visualization of large graphs. Undergraduate Honors Thesis, University of Arizona.