Computer Sciences Department

What Should Graduate Students Know Before Joining a Large Computer Architecture Project?

Shubhendu S. Mukherjee

Technical Report #1336

January 1997

UNIVERSITY OF WISCONSIN MADISON

What Should Graduate Students Know Before Joining a Large Computer Architecture Project?

Shubhendu S. Mukherjee

Computer Sciences Department
University of Wisconsin-Madison
Madison, Wisconsin 53706-1685 USA
(shubu@cs.wisc.edu)

Abstract

Large projects have become common in the computer architecture research community today. This article examines some of the things that graduate students who become involved in such large projects should be aware of.

The purpose of this article is not to advocate that a graduate student should join a large project; rather, this article should be viewed as an aid that helps a new graduate student make this decision.

1 Introduction

In recent years academia has seen the proliferation of large computer architecture projects that involve several graduate students, faculty, and research staff. The worldwide computer architecture home page[1] lists more than 40 projects with four or more researchers. This trend is also illustrated by the gradual rise in the average number of authors per paper every year in the International Symposium on Computer Architecture or ISCA (Figure 1). If projects involving only one graduate student and one faculty advisor dominated, then one would expect this number to be very close to two. However, Figure 1 shows that this number has risen from 1.6 in 1973 (first ISCA) to 3.4 in 1996. Similarly, the percentage of papers with four or more authors has risen from 0% in the 1973 ISCA to 32% in 1996 (see Figure 2 in Appendix). Similar trends can be observed in MICRO (International Symposium on Microarchitecture) and ASPLOS (Architectural Support for Programming Languages and Operating Systems). The Appendix contains the rest of the graphs. These data suggest that large projects have become common in the computer architecture research community today.

The shift to large projects, I believe, is primarily a result of the increasing complexity of the methodology used to evaluate architectural ideas. Today's microprocessor architectures have hundreds of complex components such as multiple execution units, lockup-free caches, coalescing store buffers, reorder buffers, speculation tables, etc. Adding or changing any component requires not only a detailed evaluation of the component itself, but also a detailed evaluation of the interactions of the component with other components of the microprocessor.

1. URL: http://www.cs.wisc.edu/~arch/www/

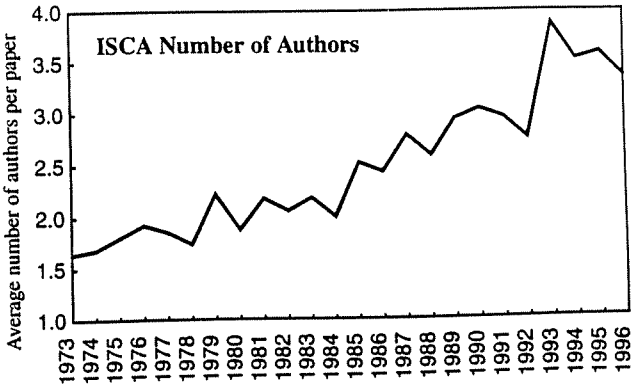

FIGURE 1. This figure shows the increase in the average number of authors per paper per year for ISCA.

Since no one has yet proposed analytical models that can capture such details, computer architects have resorted to the two other alternatives to evaluate computer architectures—hardware prototyping and simulation, both of which often require a large number of people. Hardware prototyping may require a large number of people because all details of the design must be worked out in a short amount of time.

Compared to hardware prototyping, simulators can be built much faster. However, simulators often require a large number of people because they are becoming increasingly complex, particularly with the advent of timing simulations, microarchitectural techniques such as out-of-order and speculative execution, and workloads such as operating systems and databases [3]. Simulating multiprocessors is even more complex because not only must architects simulate a uniprocessor, but also the glue (e.g., network, coherent memory bus, directory protocol, etc.) that connects these uniprocessors. Many earlier multiprocessor simulators avoided some of the complexity by not simulating the processor pipeline in detail; they only simulated the memory system. Unfortunately, recent results [2] show that this may no longer be possible with out-of-order and speculative processors.

The advent of large projects in the computer architecture research community profoundly affects graduate students. Historically, a graduate student in computer architecture paired up with a faculty advisor to do her research. She wrestled with several questions before choosing an advisor. Is her advisor a good researcher? Is

her advisor a good guide? Is her advisor easy to work with? Is her advisor well-established in the community or is likely to be so in future? Can her advisor bring equipment and financial support during her graduate career? Will her advisor allow enough freedom to pursue her research ideas? Etc.

Today a graduate student has to often pick a large project along with an advisor. Hence, she not only has to wrestle with the questions about her advisor, but also has to know the advantages (Section 2) and disadvantages (Section 3) of joining a large project. However, just knowing the disadvantages is not enough; she must also be prepared to mitigate them. A student's failure to do so can result in a lot of pain and often dismissal from a project.

I do not claim all observations in this article to be original or applicable to all large projects. Some observations are obvious, some resulted from discussions with researchers in several computer architecture projects, both within and outside the University of Wisconsin-Madison, and the rest originated from my experience as a graduate student in a large computer architecture project [1]. I would also like to add that I am a fifth year graduate student in computer architecture.

2 Advantages of a Large Project

I see eight advantages to joining a large computer architecture project. First, compared to working individually with a single faculty advisor, it is often easier to have a greater and faster impact on the industry and research community if a large number of people work together toward a common research goal. Joining forces in this manner allows a project to tackle a much bigger problem in a considerably shorter duration. Given that technology (e.g., a microprocessor) is changing rapidly, a short time frame may be critical to establish and/or prove an idea. In such cases, large projects are inevitable and useful.

Second, a graduate student can receive tremendous intellectual stimulation from other students in the same project. She can easily discuss details of her research with other project members and receive excellent feedback from them because of the shared knowledge base that exists within a project. The same may not always be possible if she works individually with an advisor, particularly if other graduate students in the same department are working on widely different topics.

Third, a large shared software and hardware base developed by project members is often available to the graduate student, which allows her to devote more of her time on analyzing her problem than developing the infrastructure to conduct her research. For example, she may be able to use several tools and benchmarks developed by other project members. This is particularly helpful when these tools and benchmarks use non-standard interfaces, so that they cannot be acquired from researchers in other universities. Additionally, a graduate student receives internal maintenance support such as rapid bug fixes and upgrades for these research tools, which may not always be possible for externally acquired tools. The downside of using internal software, however, is that it can slow down a graduate student's progress by making her dependent on other project members.

Fourth, a student receives credit for being part of a project, particularly if the project becomes famous and has a major impact on the research community and/or industry. The student will be identified with the project, which can be very useful for the student when she looks for a job.

Fifth, interactions with members from a large project can help her acquire social and professional skills required to work in a large group. This experience can be useful in the industry where projects often contain a large number of people.

Sixth, joining a large project often helps a new graduate student learn how to conduct research by watching how senior project members work. Thus, a new graduate student can learn how to do research through real and contemporary examples.

Seventh, graduate students often develop a closely knit network of peers within the same project. These peers become lifelong contacts and can be professionally helpful in many ways. Such close contacts may not always develop for students working on individual projects.

Eighth, a graduate student can often have a large number of publications if she works in a large project because she can co-author several group papers.

3 Disadvantages of a Large Project

I see eight disadvantages to joining a large computer architecture project. A graduate student must not only be aware of these disadvantages, but must also know how to work around them. Below I discuss these disadvantages and how she can try to mitigate them.

First, there is a tension between the identity of a project and the identity of a graduate student. A graduate student wants to be identified with a project, particularly if it becomes famous. Nevertheless, the graduate student also needs to establish an identity for herself in the architecture research community. For *lead* students, who shape the general direction of a project along with the faculty, this problem may be much easier to resolve because the identity of the project is often their identity. However, for other graduate students this may be a harder problem because the student has to establish her identity within the context and general direction of a project. I see no easy solution to this; she must constantly try to identify research problems that will provide her with an identity that is distinct from the rest of the project members.

Second, a graduate student may have to do a large amount of work to maintain the shared base of software and hardware. This may involve releasing software to people within and outside the project. This may bring fame; but, this may also involve a lot of extra work. A graduate student should be careful not to spend all her time doing such work because this may delay her graduation.

Third, a graduate student may not be able to influence the general direction of a large project, unless she is one of the lead students in the project. To become a lead student in a project, the student may have to join the project just when it is starting up. Hence, graduate students who want to assume the role of a lead student should look for brand new projects that faculty members are planning.

Fourth, a graduate student must be prepared to defend the goals, directions, and philosophies of the project she is involved in. I call this defending the "party line!" Typically, faculty and lead students provide the direction for the project; they are very much aware of the arguments on which the project is based upon. A graduate student, who is not a lead, must make herself aware of these reasons because she may have to defend her project in a conference or interview talk.

Fifth, although a graduate student in a large project may have a large number of papers, she may not have a large number of first author papers. Being a first author is important for a graduate student because that gives her more visibility than the rest of the student authors on the paper. How a first author is chosen often depends on the mechanics of a particular project. Often the student that does most of the work is assigned the first author of a paper, particularly if the work is related to her thesis topic. Hence, for a graduate student it is more important to get involved in a piece of work that she can lead and shape. This may imply fewer joint papers because she may have to spend most of her time working on one problem.

Sixth, in a very large project, a graduate student may not be able to directly work with her faculty advisor. Often she is advised on a regular basis by a senior graduate student or a research associate. A faculty advisor can, I believe, provide better advice than senior graduate students or research associates. To avoid being advised by senior graduate students or research associates, a graduate student should look for the student-faculty ratio in a project. Realistically, I do not believe, most faculty can advise more than four to six students at a time; so, if the ratio exceeds six, the student may not get advice on a regular basis from her faculty advisor. However, if a graduate student is involved in such a project where she is advised regularly by senior graduate students or research associates, she should try to occasionally meet with and ask for feedback from her advisor.

Seventh, in a large project papers are often written by the faculty and the evaluation (e.g., simulation work) is done by graduate students. This can happen particularly when there is a deadline for a paper submission (e.g., for a conference) and/or there is a large number of authors in the paper. This is bad news for the graduate student because she is not acquiring her writing skills. There are two ways to mitigate this problem: time permitting she can try to write the first draft and let a faculty member correct her writing and/or she can take a course in technical writing that will help enhance her writing skills.

Finally, joining a large project implies attending more meetings. A graduate student has to regularly meet not only with her advisor, but also with the entire project group and often subgroups within the project. These meetings can be counter-productive, if they primarily become social gatherings. There are several ways to make these meetings more useful. For example, people can read external papers or have project members give talks on their current research during the meeting.

4 Summary of Observations

Large projects have become common in the computer architecture research community today because of the increasing complexity of the methodology used to evaluate architectural ideas. Hence, often today's graduate students in computer architecture have to pick not only an advisor, but also a large project. This article discusses several advantages and disadvantages associated with a large project. A graduate student must not only be aware of these advantages, but must also know the disadvantages associated with a large project and be prepared to mitigate them. The purpose of this article is not advocate that a graduate student should join a large project; rather, this article attempts to discuss some of the tradeoffs associated with a large project, so that a graduate student can make an informed decision.

Acknowledgments

I would like to thank Sarita Adve, Paul Bradley, Doug Burger, my advisor Mark Hill, Jim Goodman, Jim Larus, Steve Lederman, Subbarao Palacharla, Toshi Shimizu, Guri Sohi, and T. N. Vijaykumar for their helpful comments on different drafts of this article, Kevin Theobold for maintaining the on-line bibtex references (http://www-acaps.cs.mcgill.ca/doc) that helped me generate the figures in this article, and members of several large architecture projects, both within and outside the University of Wisconsin-Madison, for a lot of interesting discussions that led to this article.

Disclaimer. All views in this article are completely my own and do not necessarily reflect the opinion of any other member of the Wisconsin Wind Tunnel project.

References

[1] Mark D. Hill, James R. Larus, and David A. Wood. The Wisconsin Wind Tunnel Project: An Annotated Bibliography. *Computer Architecture News*, 22(5):19–26, December 1994. (Frequently updated. Web location is

http://www.cs.wisc.edu/wwt).

[2] Vijay S. Pai, Parthasarathy Ranganathan, and Sarita V. Adve. The Impact of Instruction-Level Parallelism on Multiprocessor Performance and Simulation Methodology. In *Proceedings of the Third IEEE Symposium on High-Performance Computer Architecture*, 1997.

[3] Mendel Rosenblum, Stephen A Herrod, Emmett Witchel, and Anoop Gupta. Fast and Accurate Multiprocessor Simulation: The SimOS Approach. *IEEE Parallel and Distributed Technology*, 3(4), Fall 1995.

Appendix

This appendix contains five graphs that show how the average number of authors per paper has been increasing every year for ISCA (International Symposium on Computer Architecture), MICRO (International Symposium for Microarchitecture; before 1988 this was known as the Workshop on Microprogramming), and ASPLOS (Architectural Support for Programming Languages and Operating Systems). Unlike ISCA and MICRO, ASPLOS is not held every year. For the ASPLOS graphs, black dots indicate the years in which the conference was held.

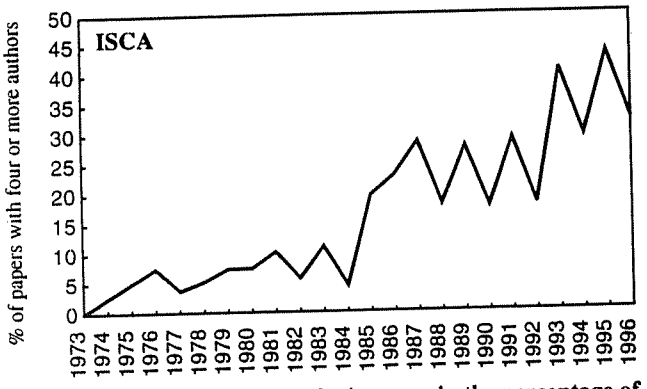

FIGURE 2. This figure shows the increase in the percentage of papers with four or more authors for ISCA.

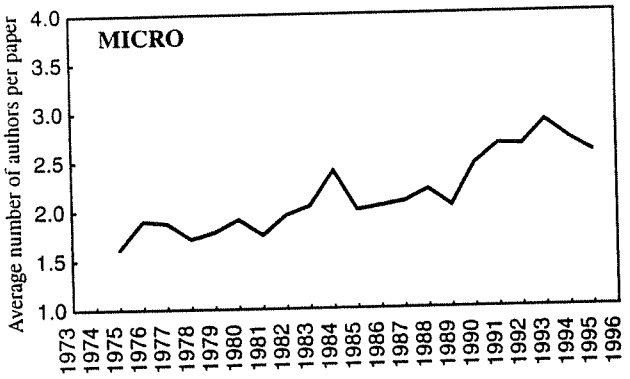

FIGURE 3. This figure shows the increase in the average number of authors per paper per year for MICRO. The first MICRO was held in 1967. Unfortunately, the data from 1967 to 1974 are missing from the graph because I did not have access to the corresponding MICRO proceedings.

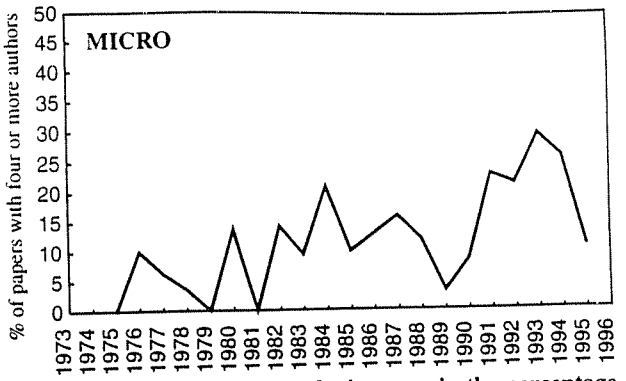

FIGURE 4. This figure shows the increase in the percentage of papers with four or more authors for MICRO. The first MICRO was held in 1967. Unfortunately, the data from 1967 to 1974 are missing from the graph because I did not have access to the corresponding MICRO proceedings.

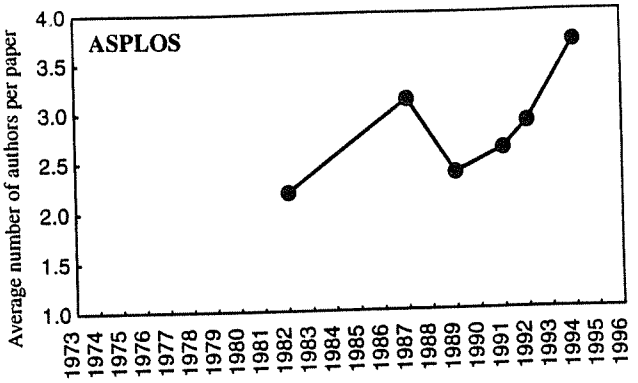

FIGURE 5. This figure shows the increase in the average number of authors per paper per year for ASPLOS. The first ASPLOS was held in 1982. The second ASPLOS held in 1987 had an unusually high average number of authors per paper, which creates the jump at 1987.

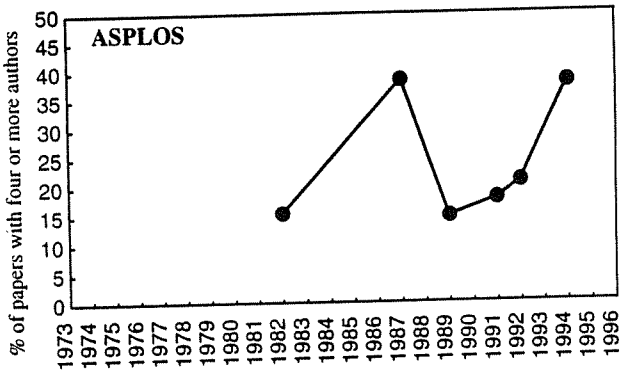

FIGURE 6. This figure shows the increase in the percentage of papers with four or more authors for ASPLOS. The first ASPLOS was held in 1982. The second ASPLOS held in 1987 had an unusually high average number of authors per paper, which creates the jump at 1987.