

Strategies for Preparing Computer Science Students for the Multicore World

Richard Brown
St. Olaf College
Northfield, Minnesota, USA
rab@stolaf.edu

Elizabeth Shoop
Macalester College
Saint Paul, Minnesota, USA
shoop@macalester.edu

ABSTRACT

The recent shift to multicore computer architecture creates a demand for all computer science students to learn about parallel computing, since software must now employ parallelism in order to take advantage of hardware performance improvements. This working group will explore strategies for introducing parallelism into CS courses and curricula, focusing on three aspects: what to teach about parallelism; how to get that material taught; and how to share materials and strategies effectively. The results of this study will hopefully spark research and further discussion within the international CS education community.

Categories and Subject Descriptors

K.3.2 [Computers and Education]: Computer and Information Science Education—*Curriculum*; D.1.3 [Programming Techniques]: Concurrent Programming—*Parallel Programming*; H.5.m [Information Interfaces and presentation]: Miscellaneous

General Terms

Algorithms, Design, Documentation, Performance

Keywords

parallelism, parallel tools, multicore architectures, curriculum, computer science educators, course material, information sharing

1. MOTIVATION

The message from the computer hardware industry is clear: for the foreseeable future, hardware performance improvements will largely follow from multiplying the number of CPU cores within each processor package, rather than constructing larger, higher-performance individual CPUs [6]. As the influential 2006 View from Berkeley white paper warns, forthcoming multicore computers that will have dozens, and eventually thousands of cores per chip [2, 3].

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ITiCSE '10 Bilkent, Ankara, Turkey

Copyright 200X ACM X-XXXXX-XX-X/XX/XX ...\$10.00.

Before this change, software performance improvements could ride with the former exponential growth in sequential processor clock speeds, but now software development must take multicore architectures into account in order to achieve substantial advances in performance [6]. To quote Herb Sutter and James Laurus of Microsoft, “Our parallel future has finally arrived: new machines will be parallel machines, and this will require major changes in the way that we develop software” [8]. This means that the CS educators must begin incorporating more training in parallelism into CS curricula, so that the students we graduate will have the preparation they need in order to create and make the best use of software designed for multicore parallel computers. The need to make such curricular changes is urgent, because the multicore revolution is already upon us, as indicated by the wide availability of two- and four-core processors in commodity computers, and Intel’s recent demonstration of a chip with 48 Pentium cores being made available for researchers [5].

This working group will explore how the international CS education community can take timely action to inject appropriate parallelism into CS curricula, in response to the onrushing shift to multicore hardware.

2. AIMS

The working group will develop strategies for bringing additional parallelism into computer science courses and curricula at various academic levels (including introductory courses) in response to the computing industry’s shift to multicore computer architectures. Group interactions will largely revolve around three strategic aspects of this issue: 1) what to teach about parallelism; 2) how to get that material taught widely and soon; and 3) how to share materials and strategies. The group’s report will summarize its findings and include recommendations intended for CS educators and others who seek to address the issue of ensuring that students are capable of developing solutions for the increasingly parallel hardware they will encounter after graduation. The results will be the opinion of the group, based on its study, which will hopefully spark research and further discussion within the CS education community.

3. THREE PARTS OF ANY STRATEGY

The three strategic aspects listed above deserve some elaboration. Below we introduce some initial questions that the group will consider when discussing each of these aspects of an overall strategy.

What shall we teach about parallelism? The group will seek content and practices related to parallel computa-

tion that are most essential for students to learn, in order to be prepared for multicore computing. What parallelism topics and concepts do students need most to learn? What should students know in order to judge when to use parallel techniques, and when not to use them? Which needed skills with parallelism already occur in standard courses? Are there parallel computational skills that every graduate should possess, e.g., applied experience with parallel loops or programming with threads?

How can we get appropriate parallelism taught widely and soon? This aspect includes at least two facets. An implementation facet pertains to how CS educators might incorporate the essential content and practices of parallelism into their courses or curricula. For example, how might such topics be integrated within existing course syllabi? What are advantages and disadvantages of particular plans, for example, consolidating parallelism topics into a single required course? Or, should nearly every CS course contain some parallelism? What kinds of support materials are needed in order to encourage the teaching of parallelism soon? Also, an advocacy facet concerns ideas for bringing about and supporting these changes expediently, such as promoting dialog, or developing arguments to justify resources that may be required to teach more parallelism.

How can materials and strategies be shared effectively? If we wish to act quickly, we must share ideas and materials in an expeditious manner. How shall we enable submission of materials and free exchange of ideas, while hiding certain details (such as answers) from students, and maintaining the intellectual property rights of contributors? How can we get these ideas and materials in the hands of educators easily? While repositories such as CITIDEL exist [1], Mitchell and Lutters found that they are underutilized by CS educators [7]. If we place materials in CITADEL, how shall we publicize their existence? Is it also important to have a forum to discuss the issues we mention above and provide summary reports on various strategies? What organization and format should be used for sharing material? Edwards et al. have proposed a format for programming assignments that we can adopt [4]. Do we need to share other types of educational materials, such as syllabi, lecture material, activities and lab exercises, rubrics, and assessment instruments?

4. PARTICIPANTS AND PROCESS

The ideal collection of participants for this working group will include persons with a variety of perspectives. Most would be CS educators interested in teaching topics in parallelism. Our discussions and conclusions will be enhanced by group members from different types of institutions and different countries. It would be important to have someone with relevant experience in industry, and/or one or two educators with expertise in computer architecture and connections to industry, who could not only take a lead on technical issues but could also portray industrial viewpoints and trends. One or more group members with administrative experience would have valuable contributions to make, particularly with issues such as advocacy. Experience with web tools and social networking would greatly inform our discussions about strategies for sharing materials. These examples illustrate the diverse group members we seek.

During the electronic phase, the group will proceed by independently exploring issues relevant to teaching parallelism

in response to multicore computing, continually reporting to the group, and formulating and sharing opinions as they go. These contributions will be collected electronically (using a collaborative site administered by the group leaders), each organized or tagged according to the three strategic aspects listed above. During the month before the conference meeting, elements of a draft report will be identified, and group members will prepare materials and position statements to be presented in group sessions at the conference, at which time the group will make final decisions about inclusion in the report draft.

5. GROUP LEADERS

Richard Brown is Associate Professor of Computer Science at St. Olaf College, and longtime Director of CS there. The architect of St. Olaf's CS major, he has created eight courses offered in that major, including a recent topics course in parallel computing. He has directed dozens of undergraduate students in over 30 cluster computing projects in the last four years. Elizabeth (Libby) Shoop is Associate Professor of Computer Science at Macalester College, where she maintains an active undergraduate research program, including several projects involving interactive web-based applications, and has also introduced a course in parallel computing. Brown and Shoop have jointly received U.S. NSF CCLI grant funding to produce materials and software for teaching parallelism within existing CS courses.

6. REFERENCES

- [1] CITIDEL: Computing and Information Technology Interactive Digital Educational Library. <http://www.citidel.org/>.
- [2] K. Asanovic, B. Bodik, B. Catanzaro, J. Gebis, P. Husbands, K. Keutzer, D. Patterson, W. Plishker, J. Shalf, S. Williams, and K. Yelick. The landscape of parallel computing research: A view from Berkeley. Tech Report UCB/EECS-2006-183, EECS Department, University of California, Berkeley, Dec. 2006.
- [3] K. Asanovic, R. Bodik, J. Demmel, T. Keaveny, K. Keutzer, J. Kubiatowicz, N. Morgan, D. Patterson, K. Sen, J. Wawrzynek, D. Wessel, and K. Yelick. A view of the parallel computing landscape. *Commun. ACM*, 52(10):56–67, 2009.
- [4] S. H. Edwards, J. Börstler, L. N. Cassel, M. S. Hall, and J. Hollingsworth. Developing a common format for sharing programming assignments. *SIGCSE Bull.*, 40(4):167–182, 2008.
- [5] M. Feldman. HPCwire: intel unveils 48-Core research chip. <http://www.hpcwire.com/features/Intel-Unveils-48-Core-Research-Chip-78378487.html>.
- [6] J. Larus. Spending moore's dividend. *Commun. ACM*, 52(5):62–69, 2009.
- [7] S. M. Mitchell and W. G. Lutters. Assessing the value of computer science course material repositories. In *Conference on Software Engineering Education and Training Workshops*, volume 0, page 2, Los Alamitos, CA, USA, 2006. IEEE Computer Society.
- [8] H. Sutter and J. Larus. Software and the concurrency revolution. *Queue*, 3(7):54–62, 2005.