Report on the NSF Workshop on Broadening Parallel and Distributed Computing Undergraduate Education

August 17 - 18, Arlington, Virginia

Organized by the Center for Parallel and Distributed Computing Curriculum Development and Educational Resources (CDER)

Introduction

Over the five years prior to the workshop, the CDER Center had designed and promoted a curriculum guideline (see http://grid.cs.gsu.edu/~tcpp/curriculum/?q=home) for undergraduate level instruction in parallel and distributed computing (PDC) That guideline influenced and was referenced by the ACM 2013 Computer Science Curriculum. The effort focused on introducing PDC concepts in the first two years of the curriculum, so that students could start learning early to incorporate PDC into their problem solving approaches. It was followed up with a series of workshops, and small grants to early adopters of the curriculum. Although successful in many regards, much remains to be done to help the discipline shift to embracing PDC education more broadly.

This workshop brought together a diverse group of stakeholders with interests in PDC education to facilitate discussion, networking, and to gather input for future efforts. The conversations were guided by three questions:

Question 1: What more can be done to broaden participation in the implementation of parallel and distributed computing topics in the first two years of undergraduate computer science and engineering programs?

Question 2: Assuming broad implementation of PDC topics in the first two years of undergraduate education, what opportunities does that preparation open for deeper coverage in upper level undergraduate and graduate courses?

Question 3: Given the current PDC curriculum guidelines and the discussions resulting from the previous questions, how should the guideline be updated?

This report describes the process of assembling the team of workshop participants, the organization of the workshop, the process used to facilitate the discussions in the workshop, and the results from those discussions. It concludes with a set of recommendations based on the points identified in the discussions.

Assembling the Team

The goals of the workshop were to gather ideas for broadening participation in uptake of PDC topics in undergraduate education, to envision possibilities for extending coverage of PDC topics to upper level courses in novel ways once a foundation is established in the first two years, and to obtain feedback on the curriculum guideline itself. These goals strongly influenced the process of searching for participants.

To address the goal of broadening participation, it was essential to have representatives from institutions serving underrepresented groups so that their experience and knowledge could inform our discussions. For the goal of envisioning ways to expand coverage of PDC in upper level courses, we needed participants who had a strong record of exploring approaches for PDC undergraduate education and teaching upper level subjects. To provide an up-to-date perspective on core topics for PDC education that should be in the next version of the curriculum guideline, we sought people conducting cutting edge research. We especially wanted people who could bring expertise to more than one of these issues. In addition, we felt it was important to have representatives from industry and government, as they are major stakeholders in PDC education.

As soon as funding was approved for the workshop and a location had been selected (late June) we began sending invitations via email. A total of fifty nine people were invited, Of those, twenty declined due to a variety of conflicts. Two did not respond, and one accepted but then had to withdraw. That gave us thirty six attendees. Of those, eight were considered organizers (four from the CDER center, four from NSF). Twenty-eight of the participants were not associated with the workshop organization. The attendees are listed below, with their organizational affiliation.

Organizers

Charles Weems, U. of Massachusetts, Amherst (CDER)
Alan Sussman, U. of Maryland, College Park (CDER)
Arnold Rosenberg, Northeastern U. (CDER)
Anshul Gupta, IBM, (CDER)
Sushil Prasad, NSF
Almadena Chtchelkanova, NSF
Amy Apon, NSF
Mimi McClure, NSF

Participants

Anyndia Banerjee, NSF
Randy Bryant, White House Office of Science and Technology Policy
Barbara Chapman, U. of Houston
Debzani Deb, Winston-Salem State U.
Akshaye Dhawan, Ursinus College
John Dougherty, Haverford College
Trilce Estrada, U. of New Mexico
Diana Franklin, U.C. Santa Barbara
Eric Freudenthal, U. of Texas, El Paso
Ajay Gupta, Western Michigan U.
Karen Karavanic, Portland State U.
George Karypis, U. of Minnesota
Dan Katz, NSF
Andrew Lumsdaine, Indiana U.
Brandeis Marshall, Spelman College
Duane Merrill, NVIDIA
Edusmildo Orozco Salcedo, U. of Puerto Rico

Cynthia Phillips, Sandia National Laboratory
Erik Saule, U. of North Carolina, Charlotte
Chi Shen, Kentucky State U.
Libby Shoop, Macalaster College
Michelle Strout, University of Arizona
Violet Syrotiuk, Arizona State U.
Michela Taufer, U. of Delaware
Dominique Theibaut, Smith College
R. Vaidyanathan, Louisiana State U.
Susan Wang, Mills College
Michael Wrinn, Intel

The attendees were approximately balanced in gender. The faculty came from schools ranging in size from a small to very large, both public and private. Many of the faculty were from schools serving underrepresented groups. In addition, three government agencies (NSF, OSTP, and Sandia) and three companies (IBM, Intel, NVIDIA) had participants.

Full travel expenses were covered by the workshop, including offers of covering child care costs if needed. In spite of that, one participant was unable to arrange for child care and had to join us remotely via Skype. That was not as effective as we hoped because the level of background noise during the discussions made it difficult to hear.

The workshop took place in a meeting room at the Arlington Hilton, adjacent to the NSF offices. Food was provided for breaks and lunch so that participants could have ongoing conversations and network with each other. That also saved time around lunch, enabling sessions to run longer. Several participants expressed appreciation for the opportunity to talk with others at lunch.

Workshop Organization and Process

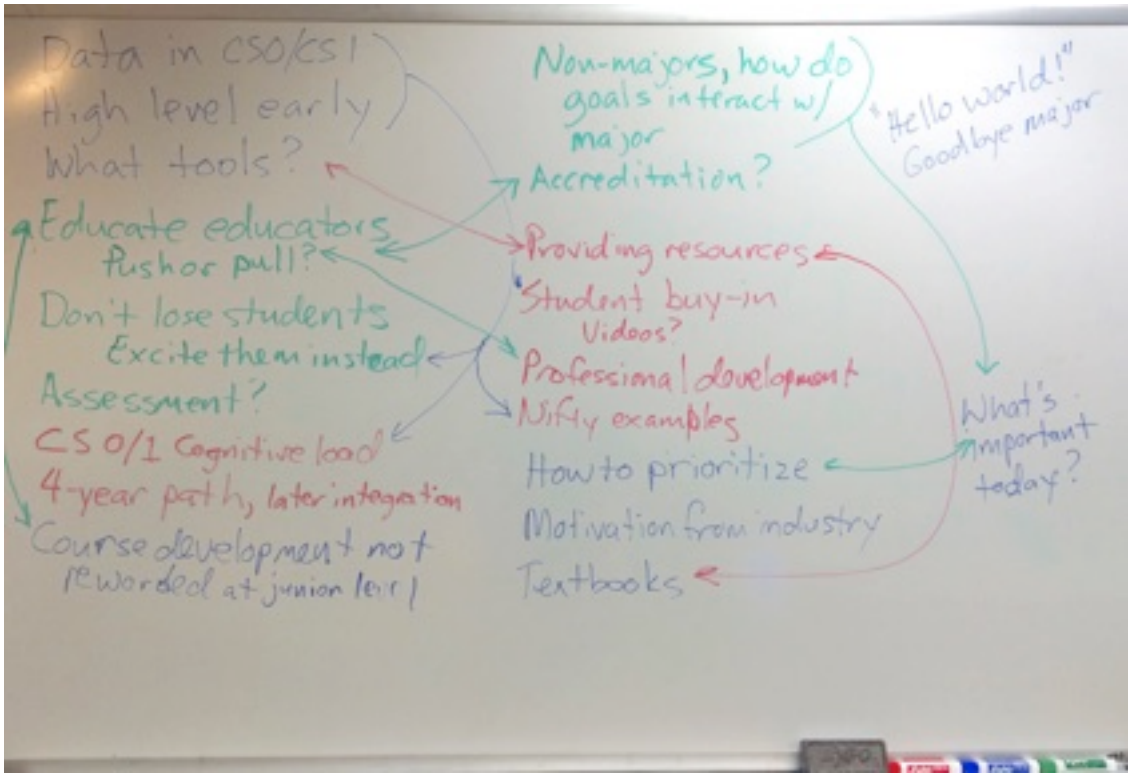
As explained in the introduction, the workshop was planned around three key questions connected with moving the PDC curriculum initiative forward. One half of each day was devoted to discussion of each question, using a World Cafe style of small group interaction. Following the first round of discussions on each topic, a followup question was formulated, based on the gathered responses, to probe deeper into an area that participants had identified as especially significant. A second round of discussions then took place for the followup question.

The World Cafe model involves groups of four people who discuss the current question at a small table for twenty minutes and make notes on a large, common pad of paper. One of the people is the table host, who helps guide the process, encouraging everyone to contribute, and to express ideas graphically on the paper.

At the end of the period, the host remains at the table, and the other three people relocate to different tables where the process repeats for a new configuration of small groups. The next round of discussion begins with the host summarizing the ideas voiced during the preceding round (with reference to the notes on the common pad). Then a new host is chosen and the participants choose whether to start a new page on the pad.

The advantages of the World Cafe model are that it gives everyone at each table time to speak and be heard. Each person can also leave a permanent record of their thoughts on the pad. As people circulate, they can both take the ideas expressed from table to table so that they diffuse across all the participants, and hear ideas (via the hosts's summary) that were coalescing in another group on the preceding round. Participants are encouraged to redistribute themselves so as to maximize the number of different people they encounter in each table they join.

At the end of a third round (after a total of one hour of discussion), there is a time for gathering of the main ideas from all the tables and general discussion. These are collected on a white board for all to see. The sheets from the common pads are also displayed on the walls of the room so that people can read the raw comments from each round. After the gathering, there is time for a break.



During a break, the organizers met to identify intersecting themes from the gathering phase and develop a follow-up question to focus on a particular aspect of the original question. Food was provided so that the participants could remain in the area outside the room and continue to engage in discussions.

Following the break another set of three rounds took place, with a final gathering and discussion phase regarding the follow-up question. After lunch, the whole process was repeated for the second question, and then again the next morning for the third question. The notes from the white board were recorded after each gathering phase. For the second question, the follow-up had participants brainstorm ideas on post-it notes that were then placed on the walls. During the gathering phase for that question, people were encouraged to rearrange the notes into apparent clusters of topics.

We were fortunate to have Randy Bryant come from OSTP and give us some opening remarks about the recently announced program to expand computer science education. Following his remarks, there was a brief introduction to the World Cafe process prior to starting the first set of discussion rounds.

The workshop schedule was as follows:

Monday

8:00 - 8:30 Continental Breakfast
8:30 - 9:00 Welcome. Opening remarks by Randy Bryant
9:00 - 9:15 World Cafe process explanation
9:15 - 10:15 Three rounds of World Cafe discussion on Question 1
10:15 - 10:40 Gathering phase
10:40 - 11:00 Break
11:00 - 12:00 Three rounds of World Cafe discussion on Question 1 Followup
12:00 - 12:30 Gathering phase and discussion
12:30 - 1:15 Lunch (provided)
1:15 - 2:15 Three rounds of World Cafe discussion on Question 2
2:15 - 2:45 Gathering phase
2:45 - 3:15 Break
3:15 - 4:15 Three rounds of World Cafe discussion on Question 2 Followup
4:15 - 4:45 Gathering phase and discussion
4:45 - 5:30 Reception discussion (light refreshments provided)

Tuesday

8:00 - 8:30 Continental Breakfast
8:30 - 9:00 Opening remarks and reflections on previous day
9:00 - 10:00 Three rounds of World Cafe discussion on Question 3
10:00 - 10:30 Gathering phase
10:30 - 11:00 Break
11:00 - 12:00 Three rounds of World Cafe discussion on Question 3 Followup
12:00 - 12:30 Gathering phase and discussion

Results of Discussions

In this section we summarize what participants identified as the key ideas in each of the gathering phases. These represent a much larger body of brainstorming from which each group, at the end of each of the third discussion rounds, would look for thoughts that had breadth of impact through interconnectedness with other ideas. Each table host would then report on the ideas around which a sense of significance had coalesced. Those ideas were captured as brief notes on the whiteboard, where they were often linked to other ideas as they were added. The following discussion expands upon those notes, based on observations of the discussions and the content of the common pad sheets. It is organized according to the original and followup questions that motivated each of the discussions.

Question 1a: What is needed to broaden inclusion of parallel and distributed computing topics in the first two years?

Work needs to go into prioritizing the material that will be in the early classes. It needs to integrate well with the existing material, open student's thinking to key concepts that can grow in later years, and enable the solving of problems that will stimulate interest in further study. Curriculum designers should keep the four-year path in mind, with more integration of topics in later courses. It is likely that a high level perspective will be necessary in early classes. For example, it may be useful to focus on the significance of data in the computer science principles course and the first programming course. Curriculum recommendations also should take into consideration the differences in goals between majors and non-majors.

It is important to get student buy-in for new material, so attention must be given to showing the benefits of learning PDC topics. We want to excite the students with nifty examples. Videos of cool applications, their impact, and the challenges overcome to make them work, may help engage students in the material. We have to be continually looking at what the important applications are today. If we just show students the equivalent of "Hello World," we will lose their interest. Industry may be able to help with the motivational aspects, but it's necessary to be careful as that can easily turn into advertising that will turn students off.

The cognitive load in the introductory courses is quite heavy, so adding PDC topics needs to be done in a way that doesn't increase it, or could even lighten it. It's important that PDC topics are not introduced in a manner that loses students by overloading them. There needs to be effort put into model course development for the introductory courses. They cannot just start integrating reworked material from junior-level PDC classes.

Educating the educators is key to enabling coverage of PDC in more diverse settings. That requires professional development opportunities, which could be delivered in either a push (going to sites to train them) or pull (holding seminars they come to) manner. It's also necessary to provide easily used resources because many educators do not have the time to develop their own or learn how to work with complicated systems. There is also a need to get the material into textbooks.

Question 1b: What is needed to educate the educators?

Money is obviously one driver of professional development. Being paid summer support to attend a professional development seminar would generate significant interest. Another option would be to run seminars at one or more major research conferences and cover the cost of attendance. For educators who may not have large research grants, that would provide PDC education, enable them to attend talks on cutting edge research, and help them to make connections with other faculty and researchers with whom they might collaborate. As part of that effort, it would be necessary to identify particular models for training that can be used, such as GENI, Software Carpentry, NVIDIA Educators Program, Codecademy, and others.

In the longer term, having education tracks or workshops associated with major conferences would allow educators to continue attending through submission of papers that would justify traveling to the conference. It was noted that simply attending this workshop gave the attendees new ideas to try, and built enthusiasm for increasing the coverage of PDC in their courses. We have heard similar comments from the EduPar and EduHPC workshop attendees. It is very valuable just to provide ways for educators to come together and share their experiences.

It will also be helpful to provide incentives to department chairs or deans in connection with efforts to educate their faculty. Such incentives could include travel grants for conference attendance, hardware and software to support PDC education, access to outside resources for supporting the PDC classes, recognitions or certifications that could be advertised, and so on.

Another way of encouraging faculty to learn about and incorporate PDC into courses is through increased sponsorship of CS-Education research for PDC curriculum development. Even a small research grant can be a strong incentive for an educator and a department to begin making curriculum changes. Current calls for proposals do not focus on PDC education, and often come with significant requirements to meet a high level of education research standards. They typically focus on comparing alternative methodologies and require involvement of education researchers, which can result in bias against curriculum experiments at smaller institutions.

Implementing curriculum changes to incorporate PDC requires access to resources that many schools do not have, beyond the parallelism present in student's multicore laptops. That also hinders educators at those schools in trying to learn about PDC tools, techniques, algorithms, and applications. Equipment is expensive both for the initial outlay and for periodic refreshing of the technology to keep it reasonably current. Resources to enable PDC education can also require extensive and costly technical support. At larger institutions, such equipment may already exist as a common resource and its maintenance can usually be folded into existing support budgets. For smaller schools, the cost of entry can be high, and ongoing support will often fall on the shoulders of the faculty who make use of the resources, both of which are impediments to professional development and curriculum changes.

The participants discussed industry relationships at length. There are pros and cons with respect to having industry involvement in education. Companies can provide access to resources, experienced professionals, and can sometimes provide investment to launch centers at schools. On the other hand, schools don't want industry representatives teaching or providing courses that are just self-serving training in vendor specific tools. There has to be a level of generality and balance.

The industry-academic relationship is complex. It is often easier for a company to focus its efforts on deans or department chairs, so that there is a more pervasive change in priorities from the top down. But success depends on the involvement and buy-in from faculty who are supportive and willing to put in the effort to make the changes with enthusiasm that infects the students. Those faculty should be identified and incentivized early in the process so that the administration can see that an industry initiative is wanted and welcomed.

Industry participation also faces a challenge of scalability. There tends to be a focus on top ten schools, because of their reputation and quality of faculty and students. However, those schools can become inbred (faculty tend to be drawn from graduates of schools in the same group), lacking in diversity with respect to smaller schools, and because of inertia associated with their size they can end up missing novel approaches. However, industry has limited resources to foster new educational efforts at a large number of schools.

Partnerships between large and small schools are potentially a valuable approach to educator development. Small schools have the flexibility for trying new ideas, and are often more focused on quality of teaching. They also present a more diverse student population, and face resource constraints that faculty at large schools may not take into consideration. Having faculty from large schools collaborate with faculty at small schools can result in an exchange of experience and knowledge that can benefit both. New approaches can be more easily tested and evaluated at small schools, and then transferred to larger school partners. The large school partner may also be able to provide access to resources that the small school cannot afford.

Such partnerships could also be more attractive for industry support because an investment of resources would go beyond a single recipient. Another leg of the partnership could be to have HPC and PDC users, such as national labs or other companies that can provide interesting data sets and applications to enrich courses. The users benefit by having access to graduates who are better trained. Special grants for such “education innovation incubators” could help to form and launch these kinds of multi-way partnerships.

Another aspect of educating the educators that was widely discussed was the need for more specificity in the curriculum guideline. It was felt that the effectiveness of the educational process could be improved if the guideline did more to specify what goes where. It was acknowledged that the original effort was meant to be descriptive, rather than proscriptive because it was felt that different programs needed to have considerable flexibility in how and where they would incorporate PDC topics. This brought up the issue that the curriculum guideline actually needs to address different populations of educators and different types of institutions if it is to be more specific.

However, providing example implementations, especially for the early courses, would help focus the efforts of educators to learn a specific set of concepts and develop their courses accordingly. There was concern expressed that otherwise it is easy to get caught up in trying to teach the flavor of the month, as PDC technology is still rapidly evolving. The guideline should offer more guidance as to how to revise the early courses in ways that have been found to make the incorporation of parallelism more natural.

It was also felt that the guideline could benefit from a simpler organization of topics, or one that is more aligned to a model curriculum so that it would be easier to advocate for its adoption and

then to implement it. The participants also recognized that the broader learning goals of the entire computer science curriculum need to be re-prioritized to make PDC as pervasive as it has become in practice beyond academia.

The group noted that HPC applications can be a source of engaging student learning experiences. But there was also a question of whether HPC has a name problem. Is it still exciting, or the most relevant? PDC topics are now at the foundation of data science, machine learning, social networks, online gaming, and other areas that are generating a lot of interest among incoming students. Thus, it may be more effective to help educators learn about PDC in those contexts rather than in terms of traditional HPC application areas.

After lunch the workshop examined a different direction for broadening the PDC curriculum. Once a school has started teaching PDC in the early courses, more of the material can be incorporated into upper level courses, with opportunities for enriched student experiences.

Question 2a: Assuming success, what new opportunities does such preparation open for deeper coverage later?

Participants noted that early coverage of PDC topics open the door to many new components or nuggets of knowledge that can be scattered across the upper level curriculum. Two subjects that were identified as potential beneficiaries of students having this background were data analytics and machine learning. In some places these would be entirely new subjects, while in others they could simply go deeper or work on larger problems.

It was felt that a variety of courses could adopt projects in data analytics in different contexts and with connections to other disciplines. For example, databases, information retrieval, bioinformatics, algorithms, networks, and scientific computing courses could all use PDC to address scaling of data analytics in different ways. A general observation was that the discipline is moving more to integrating data-driven concepts at all levels.

Having a background in PDC would better enable students to undertake interdisciplinary capstone projects. In addition to data analytics, such projects could make use of large scale simulation. Having access to PDC resources would enable students to study large or industrial scale problems, possibly across multiple courses. That also presents new opportunities for service learning work in the software engineering course. Being able to work on large scale problems would also be an enabler for new realms of exploration in entrepreneurship courses. And an early education in PDC opens up more opportunities for REUs.

It was noted that, given experience with PDC, the algorithms course can be approached much differently. In some cases, for example, serial algorithms can be a special case of parallel algorithms. It would also be possible to introduce complexity models for parallel algorithms that students would find meaningful.

Similarly the OS course can be taught differently if students have already been exposed to concepts such as synchronization, atomicity, race conditions, consistency, resource allocation, and scheduling. Obviously, a programming models class would need to cover more about PDC programming approaches, such as lock-based coherence versus transactional memory.

Although most of the discussion focused on parallelism, it was also noted that early exposure to PDC concepts would make it easier to offer an upper level distributed services course, e.g., multi-player game development, social media, financial services, and so on.

At the end of the gathering phase, the participants turned to identifying overarching goals that would be served by early coverage of PDC. It was noted that many upper level courses currently rely on projects that are intellectually interesting for the principles they demonstrate, but not necessarily engaging for the students. So one goal that is served is providing pedagogical work that is fun for the students versus just fun for us faculty. Having the ability to exploit PDC resources enables students to do work that engages the senses (things they can touch, see, hear, or otherwise feel).

Exposing all students to basic PDC concepts early means that we are engaging the other 90% of students, rather than just the people who want to specialize in PDC. Better understanding of concurrency means fewer students learn to fear concurrency. In the modern context, that is a significant enabler of career paths, and possibilities for interdisciplinary collaboration. One concluding remark was that people adopt things that reduce pain, and that is how early coverage of PDC material should be framed.

During the break, the organizers agreed that the discussion had been very general and somewhat abstract. So it was decided to ask the participants to be more concrete.

Question 2b: What are some specific examples of projects that could apply PDC, and the concepts that enable them?

Each group discussed the question and independently identified various projects. As participants moved through the rounds, the projects and concepts were further expanded. Rather than writing on the common pad, the ideas were put on post-it notes. During the gathering phase, the notes were put on one wall and participants collaboratively attempted to cluster them. It should be noted that in many cases, there were concepts that could have been addressed in multiple projects. The following lists of concepts should thus be viewed holistically, rather than being strongly associated with particular projects. The groupings are more representative of what might be emphasized in a project, even though it would also incorporate concepts from other lists. Some concepts were so pervasive that they appeared on multiple note pages, and those were distributed among the projects.

Project: Flash mob detection via Twitter feed. **Associated concepts:** Data collection and reduction. Dealing with resource limitations. Time-series analysis. Visualization. Filtering. Hooking into APIs.

Project: Multi-agent mobile/server application, multiplayer game. **Associated concepts**: Tools for debugging parallel and distributed code. Scalability at coarse and fine grained levels. Parallel task actions may be in any order. Visualization. Data organization. Concurrent data structures. Distributed data structures. Performance analysis. Load balancing. Fork/join model. Synchronization. Race conditions. Locks. Communication protocol design (Reliability, Consistency). Adversarial models. Security. Dynamic membership.

Project: Multi-disciplinary effort to identify global trends in geoscience data. **Associated concepts:** Visualization. Big data. Task and data parallelism. Performance analysis. Top ten

most embarrassing concurrent programming mistakes. Modeling. Graph algorithms. Synchronization.

Project: Seam carving (graphics). **Associated concepts:** Directed acyclic graphs. Image storage and processing. Parallel for loops. Partitioning of multidimensional arrays. Dynamic programming. Parallel recursion. Synchronization. Parallel sparse data structures.

Project: Game of life on mobile phones. **Associated concepts:** Load balancing. Distributed data structures. Point-to-point (peer-to-peer) communication. Fault tolerance. Dynamic connections.

It was also noted that it might be possible to add a programming intensive course concurrent with CS2 to motivate upper division studies. One way of viewing this was described as "Operating systems without the pain." That course could cover asynchrony and synchronization, race conditions, and divide and conquer approaches. Some people also wondered if it would be possible to work with a group such as Codecademy to develop an online course.

On the second day the workshop we looked at the existing PDC curriculum guideline with an eye toward making recommendations for the next iteration.

Question 3a: How should the PDC Curriculum Guideline be updated?

To begin with, the participants had various recommendations for the structure of the guideline itself. The current guideline assumes a basic set of courses that are fairly common across schools. However, it was felt that these courses actually differ quite a bit and that the brief descriptions of them in the guideline could lead to confusion about which local courses are more aligned with the specified courses. Thus, the guideline should incorporate a better description of the assumed courses.

Although the guideline was designed for courses in the first two years, for completeness it identifies topics that should be deferred to later courses. These are listed in the same tables but with no time allocation for the assumed courses. However, they do have recommended Bloom levels of understanding for their upper level coverage. It was felt that the Bloom levels for the advanced courses are confusing, and the guideline needs to be reorganized in some way so that they make more sense.

Participants noted that the Bloom level for any given topic would likely increase in subsequent courses. Thus there was a recommendation that the guideline show a path of evolution for Bloom levels in relation to topics over the years. That could result in a conceptual timeline indicating progression through Bloom levels for each topic. In the end, it would be helpful to have a set of Bloom levels specified for graduation, both for specialists and non-specialists.

A desire was expressed for the guideline to include example curriculum frameworks that would be suitable in different contexts. Another wish was for the inclusion of illustrative examples of the kind of coverage that is expected to achieve a given Bloom level for each topic. Another enhancement to the guideline would be to identify dependences between topics, although it was recognized that some of that varies with the depth to which a topic is covered in a specific course. In addition to the current set of assumed courses, it was recommended that the OS class should be included because it traditionally covers many topics related to PDC.

As the rounds were progressing, the organizers sensed that the discussions were again focusing at a more general level, so additional guidance was given to look at specific topics that should be deleted from or added to the guideline.

It was recommend that the revised guideline delete Flynn's taxonomy and the topic of averaging SPEC benchmarks. Specific topics that were recommended for inclusion were: Scaling and efficiency, power and energy, hardware versus software threads, atomicity, ordering and consistency, weak consistency, fences, lock-free algorithms, and accelerators as a more general model. It was also recommended that out-of-order processing should be given greater emphasis. In general, there was a sense that the guideline was biased toward parallelism, and thus there was a desire to see more specific topics related to distributed processing.

There was considerable discussion regarding the types and levels of models of parallelism that should be listed in the guideline. For example, some people thought monitors should be mentioned while others felt that they were not among a core set of models that students need to know. It was also argued that the guideline should be focused on concepts and not get bogged down in recommending particular languages or syntactic approaches. It was felt, however, that it would be good to identify a small set of simple, structured synchronization models that could be recommended, especially for the early courses.

A significant portion of the discussion emphasized that from an algorithmic complexity perspective, optimizing the amount of computation is no longer essential. Message passing, or movement of data is now more significant than computation in many contexts. That prompted criticism of the algorithms section of the guideline, which still tends to emphasize computation-based metrics. The organizers thus decided to frame the follow-up question to specifically address the algorithm topic section.

Question 3b: How should updating the algorithms section of the curriculum be addressed?

The group indicated that the number of algorithms topics in the guideline is challenging, and that the granularity of the topics is finer than other areas. Three suggestions for addressing this organizational issue were to make the algorithms list shorter, or turn it into more of a laundry list with collective Bloom level and time expectations, or to condense and cluster the algorithms topics into related categories that are more broadly characterized, with individual topics listed within each category.

A more substantial reorganization could be based on developing learning goals (such as parallel thinking) and work backward to topics that are key to achieving the goals. The participants agreed that parallel and sequential algorithms are based on different models, and saw that as an opportunity to take a different approach from sequential, which is already challenging because of the number of algorithms it tries to cover.

It was also noted that algorithms textbooks are still focused heavily on the sequential model, where computation is the metric, and that this needs a dramatic change to help students grasp what complexity means in modern modes of computation. The relevance of sequential algorithms and complexity analysis to the future of computing is a serious question. The time/energy tradeoffs between data movement and local computation (or redundant computation) are

growing in importance at all scales and should become a more central focus. Even existing parallel models like PRAM and BSP/CILK are becoming less relevant. Although the current guideline mentions power, the participants felt that it was done so in a way that was confusing.

Other suggestions for reorganizing the algorithms section included moving some topics to other areas, such as introducing matrix algorithms as a programming topic. It was felt that many of the algorithms topics could be moved to or coordinated with the programming topics area because the teaching of new algorithms is often most effective when students can experience their use in real world examples, and especially if they can contrast them with algorithms that demonstrate poor performance or even a slowdown over sequential implementations. For example, data classification and clustering could be a motivating study for a range of algorithms. In general, it was suggested that in making the algorithms section more goal driven, that there would be a shift in emphasis to algorithms that are relevant to modern data-centric problems, such as graph algorithms.

In reviewing the list of topics, it was also pointed out that their emphasis is on shared memory and high performance computing. Even with many-core systems (and it was pointed out that the guideline doesn't say anything about many-core versus multi-core systems), shared memory is diminishing in the fraction of computation that it represents. Much more computation now involves some form of message passing. That also leads more naturally into an increase in the coverage of distributed algorithms, which are growing in importance.

There were a few suggestions that were more specific. One was adding visualization of an odd/even exchange sort to the Sorting topic. And although Work and Makespan are listed as topics, it was a suggested that they might become an element of Asymptotics.

Recommendations

In this section we distill the results of the discussions to a set of recommendations for moving forward in achieving the goals that motivated each of the three topics in the workshop.

Broadening Participation

A key challenge is educating the educators. That involves the provision of training opportunities and resources, especially for those whose schools would struggle to pay the associated expenses. One model would be to offer summer training seminars that include support for the attendees. Another is to co-locate training seminars with major conferences and support travel costs. Both could be needed because some teaching faculty find it difficult to attend conferences during the school year, while those with more flexibility find it valuable to make connections at conferences and see new research. There are currently no NSF funding opportunities that would support attendees and also pay the seminar developers for their efforts, especially given that these efforts would need to be repeated for several years to reach a broad audience. New language in RFPs will be needed to enable that kind of program.

Another key challenge is providing incentives for departments to adopt curriculum changes. The original CDER grant enabled distribution of small awards (about $2500) to faculty members who proposed any reasonable level of adoption of the curriculum guideline. Those grants, coupled with coordinated equipment donations, had a significant effect. But for many schools the amount

would need to be somewhat larger to make it possible to implement such changes. The CDER grant also made provision for travel grants to enable early adopters to attend workshops associated with major conferences. Building a community of educators was seen as an important aspect of their professional development. The language in the NSF RFPs that enabled the CDER center to exist is no longer present, and there is no alternative for funding a similar effort moving forward.

Access to PDC hardware and software is another enabler of broader adoption of the PDC curriculum. The CDER center has built a cluster for PDC educational use that is freely available, and that could serve as a testbed for developing a model platform for other educational systems. But NSF should seriously consider supporting a set of educational centers in the long term to complement the investment in research centers. Such centers will need to provide a different level and model of service that should be defined by the potential users.

Partnerships between industry, large schools, smaller schools, and HPC users are a potentially fruitful, high leverage investment for both broadening participation and developing novel approaches to curriculum. A program specifically targeted to support HPC education innovation incubators would be a means of fostering such partnerships.

There is also a need to update the curriculum guideline in a manner that would include more implementation examples, and a few models of which topics can go where in various courses.

Expanding PDC in the upper division, building from early coverage

There is great potential for more meaningful work in upper level courses if students have already worked with PDC in the first two years. However, developing new approaches depends on early adopters working with those who teach upper level classes to run pilot classes. Those classes could then become models for others to adopt.

Again, it would be helpful to have a program of small grants to provide incentives for experimental curriculum and course development, along with support for the grant recipients to attend workshops to present their work. As noted previously, NSF does not currently have a mechanism through which such a program could be proposed.

The PDC curriculum guideline could be extended to include recommendations for upper level coverage. However, upper level courses are much more diverse so it would be best to develop more of a menu of topics, where each would have examples of how different Bloom levels could be achieved. Because of the greater diversity, the committee designing the extended guideline would have to be larger, possibly organized into subcommittees. While an initial guideline could help focus pilot course efforts, it should have a multi-year evolutionary process that uses the results of pilot efforts to guide ongoing revision.

Another effort would involve assembling and documenting data sets and code modules to serve as a repository of resources for use in courses. If these could also be hosted on public educational clusters, they would be more directly usable. The incubator partnerships already mentioned could be another means of developing such resources.

PDC Curriculum Guideline Revision

The curriculum guideline is now five years old, and should be revised before the next ACM curriculum revision takes place. Its revision should involve a more diverse set of contributors, and should especially seek to include experts in data-centric and distributed computing.

A revised curriculum should more explicitly define the assumed courses and be expanded to include at least operating systems among the covered courses. It should also provide more implementation guidance in the form of curriculum frameworks and specific examples of where and how the topics can be covered. Some of the guidance can come from the experience of early adopters. Creating a dependence graph for the topics would also be helpful. As an interim step, there were a few topics identified that could be deleted and added in a minor update.

The algorithms section of the guideline should have a more extensive revision to both address concerns about the focus on computation as the cost metric, and to make the list of topics more tractable through a combination of reorganization and moving some coverage to other areas.

Conclusion

Parallel and distributed computing, which employs concurrency at different scales, is the new foundation for nearly all of computer science. Many aspects of current curricula in the United States are stuck in the sequential model of computing that came to dominate the second half of the 20th century. Recent curriculum recommendations have begun to address the need for change, and some early adopters have been working to pioneer new approaches. However, it is clear that there is much left to do.

PDC also opens doors to a wide range of engaging and highly motivating course enhancements. Incorporating it should not be viewed as a burden, but rather as an opportunity for enrichment that can attract a more diverse set of students to the discipline. However, there is also a higher cost for incorporating it into coursework that could further divide academic departments along lines of size and budget, which is also likely to be biased against institutions that serve underrepresented groups. Given that PDC was previously considered a somewhat exotic and specialized subfield, many current faculty did not receive instruction in it, and thus there is also a great need for professional development to bootstrap the educational workforce's training in its use.

This workshop leads us to recommend that NSF change some of the wording in its requests for proposals to enable new programs to be supported for PDC professional development, curriculum development and workshop attendance grants, public educational PDC hardware and software resources, PDC education innovation incubator partnerships, and for revising and extending the curriculum guideline.

As part of the renewed emphasis on computer science education in this country, it is essential that those substantial investments result in development of a workforce that is prepared to use the computational tools of the 21st century to solve the kinds of problems facing us now and in the future. Thus, some of that investment should be directed toward PDC professional development, adoption incentives, enabling resources, and curriculum updates.

This material is based upon work supported by the National Science Foundation under Grant No. CCF-1546068. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.