Project for Developing Computer Science Agenda(s) for High-Performance Computing: An Organizer's Summary

Uzi Vishkin*

1 Introduction

Designing a coherent agenda for the implementation of the High Performance Computing (HPC) program is a non-trivial technical challenge. Many computer science and engineering researchers in the area of HPC, who are affiliated with U.S. institutions, have been invited to contribute their agendas. We have made a considerable effort to give many in that research community the opportunity to write a position paper. This explains why we view the project as placing a mirror in front of the community, and hope that the mirror indeed reflects many of the opinions on the topic.

The current paper is an organizer's summary and represents his reading of the position papers. This summary is his sole responsibility. It is respectfully submitted to the NSF.

2 Project Organization and Goals

The project had two stages.

- *Stage 1:* A workshop on developing computer science agendas for the High Performance Computing (HPC) program took place on March 14, 1994 in Arlington, Virginia, with nearly one hundred participants, including many leading computer science and engineering researchers in the area of HPC. The main goal of the workshop was to lead to the second, and main, stage.
- *Stage 2:* A volume with agenda/position papers, which is being published by the ACM, and is herewith enclosed. Thirty-three representatives of the U.S. Computer Science

*The University of Maryland Institute for Advanced Computer Studies (UMIACS) and Electrical Engineering Department. Partially supported by NSF grants CCR-8906949 and 9111348.

and Engineering academia and industry, listed in Appendix B below, addressed the following question: "Suppose that you are in charge of a budget of five billion dollars over the next ten years for advancing high-performance computing: What would your technical agenda be for making the greatest impact?" This volume is the main output of the project. The primary goal was to encourage researchers to think about the main issues by giving those interested in doing so the opportunity to publish their ideas. The only measures of quality that an agenda was supposed to satisfy are coherence (i.e., an agenda "should hold water"), relevance to the HPC goals, and implementability.

Having each researcher, or research group, present his/her independent opinions was of top priority. Therefore, unlike the typical work of a committee, the project's output definition did not include an agreed-upon summary. We felt that the need to reach agreement within a rigid time framework should be deferred to a later stage rather than interfere with (and possibly suppress) the independent thinking of each group of researchers.

Calls from the U.S. Congress to articulate specific and measurable goals in the area of high-performance computing, as well as the requirement of timetables and milestones for achieving those goals, triggered the project. However, the initiative for the project was voluntary and represents the view of the organizer that it is appropriate for the computer science and engineering research community to make suggestions concerning the future of high-performance computing. As noted above, this project started with a workshop and then had researchers publish their own independent agendas. The discussion and conclusions below include some suggestions on how to proceed to the next stage towards deriving a small number of comprehensive agendas that will articulate choices to government decision makers, wherever such choices are appropriate.

Support, sponsorship and organization. Support was provided by The University of Maryland Institute for Advanced Computer Studies (UMIACS). The project was sponsored by UMIACS, and the National Science Foundation Center in Discrete Mathematics and Theoretical Computer Science (DIMACS). The project was organized by Uzi Vishkin. A steering committee consisted of Richard Lipton (Princeton and DIMACS) and Uzi Vishkin.

3 Areas of General Agreement

The problem of building computer systems with increasingly useful computer power has been the "diamond in the crown of Computer Science and Engineering" similar to the way that fundamental science problems such as sequencing the human genome are at the heart of life sciences. Therefore, the U.S. computer science and engineering academia and industry should continue to be leaders in promoting attempts to tackle this problem.

The experience in the computing area has been that as computational power increases more useful applications are found, justifying considering HPC as an end in itself rather than a mean to an end. As one of the contributors put it: being a mean to many ends make the mean an end in itself. However, the extent to which an HPC program must include "deliverables" on any particular list of "grand challenge" applications (as part of HPC) was not an area of unanimous agreement among contributors.

A transition from serial programming and systems, which currently completely dominate the computing world, into (having also) ubiquitous parallel programming and systems will probably be the biggest revolution in the history of computing since its early days and is therefore of vital interest to the general computer science community. In reality, however, revolutions are implemented over time in a step-by-step evolutionary manner and require planning.

A strong case for the continuation of exploratory research that will make general-purpose parallel computing widespread was made in the position papers.

There was also general agreement that the programmability of parallel systems, their performance, and the portability of application software across a variety of parallel systems are some of the key issues facing the field. The current state of affairs is sometimes referred to as the "parallel software crisis". For more on the state-of-the-art with respect to available parallel machines we quote B. Smith. *General-purpose computers have these important attributes: (i) They display adequate performance on any program; (ii) they require minimal programming effort to achieve it; (iii) their I/O capabilities match their computational power; and (iv) they use a "standard" scalable programming model. Today's high-performance parallel computer systems are clearly not "general-purpose".* The quote implies that the parallel machines are not yet here.

The case for parallel computing has always been based on the need for growth in performance. To understand the role that parallel computing can play in responding to this need it is important to understand various sources for such growth, as well as their relationship to parallel computing. For instance, microprocessor performance has grown at a factor of nearly 2 per year in the last few years. It is clear that due to fundamental physics limitations, such growth cannot continue for too long, and some suggest that it is already slowing down. Strategies for advancing towards general-purpose parallel computing, as well as, the case for doing so, depend in many respects on the rate of this growth. In the event that *this growth comes to a halt or decreases dramatically*, parallel processing becomes an obvious choice for further growth in performance.

However, it is important to notice that the case for parallel computing is very strong even if *this growth continues for several more years*. To understand that, we observe that the scope for competition among computer system (or, hardware) vendors has becomes very limited. For instance, it is remarkable that the factor of difference in performance between some of the least expensive computers and the most expensive ones on some applications dropped from 10^4 in the 1970's to less than 20 today. So, it appears that resorting to parallelism is a good way for such a vendor who wants to obtain a competitive edge. The other side of the same coin suggests that if some architecture designs will include an increase in parallelism, staying with serial computing will probably become a sure way to lose the competition! For this argument to hold parallel computing should be scalable with respect to the continuing improvements in microprocessor performance and other potential sources for growth in overall performance. To sum up, *developing scalable parallel computing will* *contribute to the industrial competitiveness of the society that does it first.*

Finally, the title of this section should not mislead the reader into believing that every participant in this project would have written the section in the same way, or would even agree to every statement in it.

4 Main Conclusions

The conclusions listed below all relate to necessary conditions for the successful implementation of HPC.

1. Continuing exploratory research on algorithms, applications, computer systems, compilers, and programming languages.

2. Encouraging the mainstream computer system building industry to become actively involved in HPC. The main difficulty here is that the transition from serial programming and systems into ubiquitous parallel programming and systems may take from five to ten years - which is too long for the industry to be actively interested in. However, industry is likely to become actively interested in tasks that can be accomplished in two years or less, and it is a major planning problem to break the transition into parallelism into relatively short-term steps.

3. Providing an environment where independent software vendors who are interested can survive economically.

4. Items 2 and 3 require intensive deliberation on ways to advance from the current exploratory research stage to a "dual track" combination of exploratory research on one hand and a "planned revolution" on the other.

5. Securing good use of the taxpayers' on-going investment in HPC should be of the highest priority. It is of crucial importance that application domains of HPC will include the newly-defined National Challenges (e.g., the realms of education, digital libraries, health care, crisis management), enhancement of the information superhighway (which will require much stronger computing power, e.g., by information servers and for network control), and applications of commercial interest, such as data bases, electronic commerce, and real-time animation, as well the Grand Challenge scientific applications.

6. Develop gradual (i.e., evolutionary as opposed to revolutionary) pilot plans (or agendas) to implement a transition into ubiquitous general-purpose parallel computing, since it is extremely unlikely that such a revolution will occur without careful planning. An important objective of such plans would be to articulate options for action to government decision makers. Any plan for HPC amounts to reaching some balance between the overall costs for hardware and software. A first step should be to articulate the main schools of thought in this area, and to let them develop further. In Appendix

A, an attempt is made to briefly identify those schools of thought. Such a gradual plan (or agenda) should:

(a) Lead to narrowing the gaps between the state of knowledge and understanding (e.g., theoretical models which offer methods, measurements, robustness, and ease of programming) and real machines.

(b) Create robust platforms for more effective exploratory research.

(c) Create robust platforms for independent software developers to work on.

(d) Have the commodity processor evolve into functioning better in a multi-computer, perhaps by updating benchmarks (such as SPEC).

(e) Respond to the immense task of educating millions of programmers to adapt themselves to parallel programming; concrete tasks may include promoting the understanding and development of

- easy ways for parallel programming, and
- education "to think in parallel".

Also, it is important to have widely accessible computer systems to enable extensive training in parallel programming.

(f) Meet free market criteria for limited government involvement. As an example, any plan for subcontracting a task to industry should not lead to a situation where the government ends up picking winners and losers.

5 The Moon Shot Analogy

I added in the last minute the present section to the introduction chapter which I wrote for the edited volume. However, the contributors could not relate to it since they did not see it before submitting their papers. To stimulate debate at the NSF, I would like to repeat this section here.

We draw an analogy between HPC, and its objective of achieving ubiquitous general-purpose parallelism, and the U.S. space program, announced in 1961 and accomplished in 1969, to put a person on the moon ("the moon shot program"). Like any analogy, this also has its weaknesses, but we must draw analogies from other fields of science and technology since the field of Computer Science and Engineering has never before faced a challenge whose magnitude is similar to HPC.

Imagine the space program in 1961, where various vendors are developing their own products and are competing for selling them to universities or research laboratories or universities funded by by government funding agencies in rounds of three-year time windows. How successful would have this been in putting a person on the moon as early as possible?

While, we do not have the tools to compare the magnitude and complexity of the space program with HPC, we feel that at least in one aspect the HPC is more challenging. Recall that the space program was carried out by a small number of exceptionally skilled professionals (Wernher von Braun's leading team included a hundred people). However, the involvement of the masses was rather passive (essentially, watching the accomplishments in the media). While we do know how to compare the skills needed to lead HPC with the space program, the big difference with respect to HPC is that numerous programmers will have to update the way in which they perform their job. This active participation of so many people, not all with an advanced degree in computer science, should be taken into account in the implementation of HPC.

Questions: Is this analogy helpful? if yes, does it mean that HPC needs a more strict plan? how much time will be needed for its completion? if more than three years, then doesn't it mean that government must play a crucial role? what should be the role of industry? what should be the role of the academic research community? how should the implementation of such a plan be managed? who should do it?

6 Epilogue

We intentionally did not touch on many issues, such as how to bring about a situation where a search for computer science agendas for HPC continues and specific agendas are developed (for the purpose of helping articulate options for action to government decision makers). We feel that having the debate run by leading academics, who are free of the economic conflicts of interest as present in industry, will be a useful safeguard to guarantee that the taxpayers will get their investments' worth. However, we avoided discussing that since the current structure of the academic reward system and government research funding makes it very difficult for computer science generalists, who could have developed specific agendas to operate.

Finally, government funding of research should continue to search for ways to develop independent thinking on strategic directions among academic researchers especially today where there is growing pressure to tie academic research to short-term results. HPC may be the best example of strategic long-term applied research in computer science and engineering and one of the best examples overall.

7 Appendix A: Schools of Thought

Based on the position papers, it appears that current thinking can be classified into: (i) Hardware-centric; namely, focus on parallel systems that can be built in the short-term using relatively inexpensive hardware, in the hope that the application software industry will follow. (ii) Software-centric; namely, focus on possible ways in which the development of application software will be the easiest in terms of programmability. Such an approach will not be limited by available, relatively inexpensive hardware, but will rather call for feasible hardware, which may currently not be available, to be developed. (iii) Some middle ground between these two extremes.

Compared to the other approaches, a hardware-centric approach is likely to result in the least expensive hardware. Such an approach implies the lowest cost in the short-term and since systems are built by people who must have given a great deal of attention to hardware issues, a hardware-centric approach has had a natural appeal to them. Apparently, more parallel machines fall into this classification than into the others. The "parallel software crisis" makes the question of whether satisfactory development of application software will follow, even more acute. A software-centric approach is likely to result in the lowest cost for the development of application software (possibly alleviating the "parallel software crisis"). The hardware costs may be higher, at least in the short-term. Without government subsidies for the short-term hardware costs, the implementation of such an approach may be difficult to start up, even if the case can be made that a plan along these lines is likely to bring these hardware costs down later. It is unclear how the three approaches would rank with respect to the cost of developing system software. It is also interesting to study whether the third approach will enjoy the benefits of the "pure" approaches, or rather, suffer from weaknesses of both. This discussion shows that an extensive debate and better understanding appear to be necessary conditions to enable consideration of some options, which may turn out to be more attractive than the ones that otherwise would have been chosen. Such a debate will also guarantee the wide support needed for successful implementation.

8 Appendix B: List of contributors

Bowen Alpern and Larry Carter - IBM; Marco Annaratone - DEC; Guy E. Blelloch, Bruce M. Maggs and Gary L. Miller - CMU; Jerome A. Feldman - ICSI; Geoffrey C. Fox - Syracuse; Dennis Gannon - Indiana; David Gelernter - Yale; Phillip B. Gibbons - AT&T; Michael T. Goodrich - Johns Hopkins; Alan Gottlieb - NYU; John L. Hennessy - Stanford; Danny Hillis - TMC; Susan Flynn Hummel - Polytechnic; Intel Corporation (Robert L. Knighten and Timothy G. Mattson); Joseph JaJa - UMD; Leah Jamieson, Susanne E. Hambrusch, Ashfaq A. Khokar and Edward J. Delp - Purdue; Laxmikant V. Kale - Illinois; Richard M. Karp - Berkeley (Foreword); Ken Kennedy - Rice; Peter H. Mills - Duke, Lars S. Nyland - UNC , Jan F. Prins - UNC and John H. Reif - Duke; Steve Nelson - Cray; Franco P. Preparata - Brown; Abhiram Ranade - Berkeley; Umakishore Ramachandran and H. Venkateswaran - GA Tech; Vijaya Ramachandran - UT Austin; Arnold L. Rosenberg - U of Mass.; Joel Saltz - UMD; John Savage - Brown; Paul B. Schneck - MITRE; Sumit Ganguly - Rutgers and Avi Silberschatz - UT Austin ; Burton Smith - Tera; Quentin F. Stout - Michigan; Uzi Vishkin - UMD (Introduction).