Reviews

Enabling Technologies for Petaflops Computing, by Thomas Sterling, Paul Messina, Paul H. Smith. ISBN 0-262-69176-0 (paperback), 1995, $ 26.95, 200 pp. Available from The MIT Press, Cambridge, MA.

This book is a workshop report. Some interesting people were invited to come together to consider enabling technologies from which Petaflops computers might be built. The attendees heard an invited talk by Seymour Cray that was both excellent and truly visionary. He emphasized the need to learn how to use components on the nanometer and smaller scales. Although I believe he is absolutely correct, it does seem that his remarks, while well received, were more or less ignored thereafter. This should not be so surprising; the general computer industry is mostly preoccupied with the exploitation of mass-produced parts, and not with the careful system design of balanced supercomputers.

Another keynote given by K. Likharev from SUNY at Stoneybrook covered some issues related to superconductivity as an enabling technology for superfast computation. Again, the computer industries are very busy doing things that they have always done, so confronting tough things like micro manufacturing or superconductivity is considered too far from the paths most of the manufacturers are presently taking. So the workshop while being aware that such things exist, did not place its main emphasis on their use. Initially, much of the advanced developments in computing were inspired by what was going on in supercomputers. This is decidedly not true today. The machines now available are OK and up to small clusters are being used to work on those problems people first posed in the 1960s. New supercomputer problems are not being attacked as intensively as the old ones once were; the envisioning space is simply not conducive to such speculations.

Received February 1996
Revised May 1996

ISSN 1058-9244/97/$8
Scientific Programming, Vol. 6, pp. 395–397 (1997)

Practically every idea for new computer designs gets mentioned somewhere in this workshop report, so no criticism of this or that individual feature can be given unequivocally. It seems though that the greatest reliance is placed on massive parallelism. This reviewer believes that this is the most likely way to be not only wrong, but totally wrong. The evidence for achieving real Petaflops computing through massive parallelism is just not compelling, nor is it likely ever to be.

Chapters 1–3 summarize the workshop efforts and Chapters 8–10 offer some justifications why such a study is appropriate now. The other four chapters (Chapters 4–7) go into substantial detail in examining relevant enabling technologies, including software, and the possible uses of such (Petaflops) computers.

These chapters cover applications, device technology, architectures, and system software. I will provide just short descriptions of the contents of these chapters; anything else would make this review far too long.

One initial comment is that the conclusions of each of these chapters seem to be mostly independent; by that I mean, they are not to be taken as interconnected aspects of a system design. The report mentions practically all currently known technologies and methods. There was no attempt to do a system design for such a computer to show how each part contributes to the whole.

Another comment: when the discussion swings around gigaflops and teraflops, it is very hard to believe the naivete and glibness that seem to ooze from parts of the book. For example, the report hints that such performance is already common. If there was a real teraflops computer, why, they would simply gang 1,000 of them together to get a Petaflops computer. Simple, but if that won't work, one can get the same effect by ganging 1 million gigaflops computers together. The report further estimates this can be done in the next 20 years. Anyone who believes such stuff is a hot prospect for the bridge I want to sell.

Chapter 4 has some relevant discussions on what sort of problems might be done on a Petaflops computer. It is the most interesting chapter in the book: there are some excellent discussions covering program structures, but it adds little to the general truth that no matter how fast we make a computer, one can always define problems beyond its capabilities. This is like a corollary to Parkinson's law, "There is never any extra time because the work expands to fill the time available for its completion."

However, Chapter 4 also has good discussions about things worth computing in biology, chemistry, physics, and other relevant topics in the commercial, governmental, and societal areas. There is no question that they have a selection of do-able problems, but not much consideration is given to identifying those that the nation is actually willing to pay to have solved.

Chapter 5, also a good chapter, is on device technology. It contains performance projections for three kinds of technologies: semiconductors, optical components, and superconductors. A major excuse for this kind of extrapolation is that it is merely a continuation of the past: the computer industry wants us to accept their ever smaller, faster, and cheaper devices, however, unbalanced they are. Since about 1950, the cumulative factor of improvement is between a hundred thousand and one million. Depending on where you peg today's compute speeds, getting to Petaflops will require an additional speedup of (only) a thousand to 10 million times. The chapter notes, somewhat parenthetically, the device interconnect problem: ever more elements packed into ever tinier spaces, yet it blithely assumes such growth is possible. Even if we accept the possibility, a better question is, can or will the nation make the much more demanding investment especially considering our love affair with small, cheap computers working on problems largely left over from the 1960s?

Chapter 6, on architecture and systems, contains some conclusions that ought to bring readers to the edge of their chairs. "Petaflops systems are achievable in about 20 years; silicon technology can satisfy the majority of requirements if it continues its present rate of improvements; the Petaflops machine will rely heavily on technology developed for the commodity [my term] market." In the past, this has been highly unlikely; if you're in trouble with a system based on small computers, getting more small computers will not solve the problem. These conclusions notwithstanding, they note that more research is needed; and so on. The chapter contains a long list of metrics and limitations intended to assist in the evaluation of three candidate architectures: global shared memory, a network of microprocessors, and a design containing processors in the memory (PIM). There are additional discussions on the impact of device technologies, especially memories. One comment was rather arresting: "The parts count . . . is quite manageable, and indicates that in 20 years the Petacomputer will cost about what a supercomputer costs today." Elsewhere in the report are estimates of more than $100,000,000. What were they smoking?

Chapter 7 has a discussion of the needed software technology and tools. This is a fun chapter and best read after consuming at least a half bottle of decent Pinot Noir. This will save you from having to jump up after each paragraph and declaiming it's correctness – in part. You can remain seated and holler."Right – but!" Surely everyone agrees that beyond the inspiration of a seminal application, the role of software is the most important ingredient in the successful development of any computer system. So, what are these "buts"? The authors say very clearly such things as, "Current software and tools are not adequate for carrying us to the level of Petaflops." Right! "The level of investment in software and tools is insufficient for solving such problems, particularly parallelism" and, the most dreaded comment, "More research is needed." Right!

Other remarks in a similar vein can be found in this chapter. One of the most amusing estimates is that the Petaflops computer requires at least 400 terabytes of memory, implying at least 50,000 memory chips (80 gigabytes per chip? What do they know and when did they know it?). Also, to off-load this memory in a "realistic" time will require 100,000 disks. Whew!

What is less explicit is a recognition that the previous 20 years of "research" have given no hint of a general solution to the parallel programming problem, and that practically all our energies have been dissipated in broadening the computer markets and giving everyone a desktop machine that does word processing and very slow I/O. Moreover, as the computer business increasingly becomes commercial, the factors that aided its early growth have largely disappeared behind a curtain of spurious patents and copyrights. "More research is needed," is true but it must be done quickly if that Petaflops computer is to be ready in 20 years.

As already noted, the report stresses that the only way to achieve Petaflops performance is through massive parallelism. There is no confession that current attempts are not real successes yet.

The same is true about current algorithms: it is not clear that they represent the best way to use such machines. Entirely new computational models will almost certainly be needed in the future. Clearly, some thought on how to fit new architecture and algorithms together is something whose time has come. Such developments are not getting nearly enough attention, and there is no way to ensure that suitable algorithms will be available when or if Petaflops computers are available (this is my opinion).

Even though some problems have been made to run on "massively parallel" computers, there is no guarantee that

future problems will fit the same architecture. Clearly, what we today call massive is nothing compared to what will be required in producing a Petaflops system.

It might be sobering to recall some performance numbers. The nominal goal of this workshop was to recognize technologies from which a machine system could be constructed that would execute 10^{15} floating point operations per second. While there are always exceptions to the rules, generally, a floating point operation requires two 8-byte operands, which are combined to produce one 8-byte result. These 24 bytes represent some portion of the overall memory traffic. Yes, there are special floating point hardware units and all sorts of pipelines and vector units and other ways to perform such arithmetic, However, if we are to have 10^{15} floating point operations per second, on the average each flop has to complete in 10^{-15} s. – so on average, we have to move 24 bytes in a femtosecond!

How far does light or any signal move in 1 femtosecond? Well, it's no more than 0.3 μm. But also, after the signals arrive, they must be combined, and results possibly saved. To oversimplify, you have to get close: that's what Seymour Cray said and has been saying for years. A lot has to happen in a short time in that small space, but in the stated opinions of the participants, there are ways to do it. I claim that these are less than obvious, but it just doesn't seem likely to come from mass produced, commodity parts.

It is both strange and typical that the actual limit on performance in most systems gets almost no attention in the report: I/O. There is just one paragraph (p. 168 ff) devoted to I/O scaling. This is clearly a tough problem, and not many people seem to be interested in it because it's hard to do. This makes me remember the old aphorism: "If you only have a hammer, everything looks like a nail."

So, what finally, can be said about this book? I believe it's worth reading; it contains some good ideas. I liked particularly the comment (p. 150): "Although committees can extrapolate trends with some success, the 'break the mold' ideas that will allow us to reach the Petaflops goal will come from highly motivated and prepared individuals, not from committees." That's just right.

Here are some other truths that don't appear to have been changed by this report:

1. Making a parallel Petaflops computer out of a few (< 100) processors may be possible. Scaling up to larger numbers is highly questionable.
2. Nobody knows how to "do" parallel programming.
3. Trying to build a Petaflops computers out of commodity parts is naive.
4. Putting a large number of small machines together will not lead to a Petaflops computer. It seems that too many people still believe the idea that is one cat can catch one rat in 1 min, we can catch more rats faster using 1,000 cats; or 100,000.
5. The present computer industry, dazzled at it is by the mass marketing of small computers, will never produce a Petaflops computer.
6. It is, finally, not clear that the main lesson of computer design is suitably recognized: any computer system is only as fast as its slowest part.

It is true that some scientific problems have been fitted into forms such that they can be solved on some number (mostly small) of parallel processors. But generally, it's a small number of small processors. The speeds reported are in the order of gigaflops, and usually don't include I/O. Even if these claims were realistic, there is no evidence that such methods scale up another factor of 1 million times to get to 10^{15} floating point operations per second.

The problems that need to be solved to produce a Petaflops computer, must be recognized as more, much more, than mere engineering challenges and scale-ups. Fundamental issues must be resolved; indeed entirely new computational schemes may be needed, and these are not likely to result just because some well-intentioned bureaucrats snap their fingers.

I believe that the development of a Petaflops computer is largely an unprecedented problem that will be solved, but there is no preexisting basis on which to build. In this, it is similar to the production of the first atomic bomb. In the event, the solution depended on such fundamental ingredients as very good people, very little management interference, and lots of cash. My reading is that presently we are in an era of extensive management meddling and little cash, although we do have some very good people.

Finally, I cannot end these remarks without commenting on the loose quality of the editing done by the publisher. I found many small errors, both syntactic and semantic. The press considers it more important to get books speedily published than to worry about such things as proper English. This reminds me of some Microsoft policies. I say that's a cop out, or at least, a bad attitude.

George A. Michael
Lawrence Livermore National Laboratory
P.O. Box 808
Livermore, CA 94550, USA