The Transition to Massively Parallel Computing Within a Production Environment at a DOE Access Center

Michel G. McCoy

This paper was prepared for submittal to the Conference on High Speed Computing
Glenden Beach, Oregon
March 29 - April 1, 1993
DISCLAIMER

This document was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor the University of California nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial products, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or the University of California, and shall not be used for advertising or product endorsement purposes.
THE TRANSITION TO MASSIVELY PARALLEL COMPUTING WITHIN A PRODUCTION ENVIRONMENT AT A DOE ACCESS CENTER

Michel G. McCoy
National Energy Research Supercomputer Center
Lawrence Livermore National Laboratory

Introduction

Massively Parallel (MP) Computing has its religious aspect, and from that perspective, it is seen as sweeping the old faiths before it. It is a confrontation which so far has been indecisive; the old gods live yet and serve well. The sequential machines are functional, even dominant, and have not been abandoned; yet there is this nagging sense of senescence. The research community, gathered at the mount, has witnessed, if not the multiplying of the fishes and the loaves, at least enough to create an interested, if not blindly faithful, following. In the realm of computing it is not faith but reality that ultimately writes history, and this reality will be defined by the evolution to and the appearance of the production environment.

In contemplating the transition from sequential to MP computing, the National Energy Research Supercomputer Center (NERSC) is faced with the frictions inherent in the duality of its mission. All along, there have been two goals, the first has been to provide a stable, serviceable, production environment to the user base, the second to bring the most capable early serial supercomputers to the Center to make possible the leading edge simulations. This seeming conundrum has in reality been a source of strength. The task of meeting both goals was faced before with the CRAY 1 which, as delivered, was all iron; so the problems associated with the advent of parallel computers are not entirely new, but they are serious.

Current vector supercomputers, such as the C90, offer mature production environments, including software tools, a large applications base, and generality; these machines can be used to attack the spectrum of scientific applications by a large user base knowledgeable in programming techniques for this architecture. In contrast, parallel computers to date have offered less developed, even rudimentary, working environments, a sparse applications base, and forced specialization. They have been specialized in terms of programming models, and specialized in terms of the kinds of applications which would do well on the machines. Given this context, why do many service computer centers feel that now is the time to cease or slow the procurement of traditional vector supercomputers in favor of MP systems? What are some of the issues that NERSC must face to engineer a smooth transition? The answers to these questions are multifaceted and by no means completely clear. However, a route exists as a result of early efforts at the Laboratories combined with research within the HPCC Program. One can begin with an analysis of why the hardware and software appearing shortly should be made available to the mainstream, and then address what would be required in an initial production environment.

What Has Changed?

Changes in three areas have altered the views of supercomputer centers; these areas are hardware, software environments and user perceptions and requirements. The MP computer of today offers the advantage of raw processing power. That is, it offers much higher peak speeds than what is
available on conventional sequential machines. By the middle of 1994, 150-400 GigaFlops on 1000-2000 CPUs with 64-128 MegaBytes of memory per compute node is promised; this represents peak speeds two to three orders of magnitude greater than a single CPU of a C90. This can be seen in the graph below and on the following page describing the evolution of peak speeds and memories of supercomputers.\(^1\) Beyond raw microprocessor power, VLSI has provided far better communications and interconnection networks with high bandwidths. This combination of improvements represents the genesis of what could be described as architectural balance. Some of the very latest offerings are being touted, not so much on the basis of peak speed as on the basis of balance, that judicious combination of capabilities involving peak CPU speed, bandwidth to memory, interconnect bandwidth, global communications, latency hiding and synchronization. This should enhance the efficiencies of these machines, increasing the number of applications in which they can dominate or compete with classical vector supercomputers. However, from the perspective of the computer center, the critical software capabilities must be there too; this time there is going to have to be more in the box than iron. This brings us to the second area of change.

That the limitations of inadequate programming models and diagnostic software are falling away is evidenced by the growing body of results from various disciplines that have turned to parallel machines. We present a few examples from a variety of disciplines. In the area of quantum chemistry, it is necessary to determine the Hartree-Fock wave function, an approximation to the electronic structure of the molecule studied.\(^1\) Construction of the large set of integrals necessary for the task is easily parallelizable; however, a step which was not initially understood to be parallelizable was the building of the so-called “Fock matrix.” Performance gained through parallelization of that difficult step, essentially completing the parallelization of Hartree-Fock calculations, has resulted in overall efficiencies of about 60% on 500 processors of the Intel Delta.

Other examples can be cited. For instance, early MP work in the areas of climatology and plasma physics is showing promise. In the latter, aggressive efforts to understand plasma turbulence
Evolution of Main Memory Sizes

Massively Parallel estimates based on 1000-node machines or their equivalents. Projections based on manufacturers' estimates.

utilizing both particle-in-cell and gyrofluid approximations are being undertaken within the HPCCP Numerical Tokamak program. One can get a sense of the optimism in both of these areas from observation of the details in the ocean calculations or the gyrofluid calculations recently executed at the Advanced Computing Laboratory. 

This progress is not invisible to the computational physicists, who are sensing that early access to parallel machines could become a critical issue in the novelty of their research.

This brings us to the third factor encouraging computer centers to move in the direction of highly concurrent computing: the human side of the problem. Given the current rate of evolution of computing environments, by mid-1994 a number of vendors should be offering initial (early) production environments. Over the next year or two, these should evolve into full production environments. An initial production environment is probably not a bad place in which to develop an initial production code. Nor is it a bad place for researchers to develop necessary insights.

What the users realize is that limited access to high performance parallel machines will mean limited fuel to stoke the full production computers of 1996. It will mean limited applications to run production, and it will mean limited numbers of programmers schooled in the new programming techniques. The production environment is far more than the machine and the software cemented to it; it is also the applications base running efficiently and the research scientists who have developed an affinity for the machine. The lack of even one of these elements eliminates balance and creates inefficiency. The 1994-1995 time period will be wisely used if it is viewed as the critical code development link in a chain of events leading to full production parallel computing in 1996. Approaching the problem of evolution as an interdependent hardware, software and human equation will be a realistic route to a full production environment.

In summary, what has driven the change from the conservative viewpoint of centers such as NERSC, is a combination of improvements in the hardware and software which is producing dramatic results in applications. These results are in turn driving the users to demand access to powerful platforms to allow the timely development of applications in order to further research.
Two Major Debates

In order to attempt a definition of constituents of a production environment for a center like NERSC, the two major debates that have engaged the user community from time immemorial must be understood.

We will begin with the capability-capacity debate, and its variant, the workstation-supercomputer controversy. Since NERSC machines are typically saturated, there is no aggregate advantage in multitasking, if one averages over the entire workload. Consequently, the environment has typically not been tuned to favor multitasked jobs. The majority of the jobs on the machine are capacity jobs running at most a few hours, utilizing a small fraction of the total memory of the machine. One could argue that this is what the users desire; however, this rich capacity mix might be a response mechanism—the user chooses problems accessible to the computing environment. Had the machine available considerable unused capacity, the number of capability jobs could increase considerably.

In fact, as evidence of the need for far greater capability, a group of ER users and advisers recently completed an analysis of requirements for supercomputing. Computational requirements arising from a large range of ER disciplines were solicited, and the algorithms described spanned the spectrum of numerical techniques. It became clear that researchers in these fields felt strongly that a two-order of magnitude increase in peak speed over a single processing unit of a C90 would allow a qualitative jump in the character of the physics studied. Two orders of magnitude would allow a transition from routine 2-D to routine 3-D modeling for many problems, or at a fixed dimensionality, a transition from poorly resolved problems with limited physics to highly resolved problems with complete physics. The panel concluded that there is a range of problems which will require many hours for individual runs to complete on the MP machine.

On the other hand, many users are running shorter problems, which we are calling capacity mode calculations. A number of these users have decided to run their problems on the workstation. The viewpoint was that problems actually finished faster on the workstation because there was no contention for a saturated resource. Taken to its extreme, this argument leads to the acquisition of workstations instead of supercomputers. However, while a capability machine can run capacity jobs, a workstation cannot run capability simulations. As was observed by Brooks, since many jobs are capacity jobs, the machine is significantly funded by those interests. The question of how to serve the needs of capacity users within the environment of a Grand-Challenge-capable machine must be addressed.

The source of the second debate, surrounding batch processing or interactive computing, is partly historical, but primarily fundamental. There is a basic difference in the way that molecular dynamists or lattice gauge theorists might approach computational physics and the way climatologists and plasma physicists might approach it. In some cases, the QCD theorist may know the answer to the problem and wishes to test the underlying theory whereas the plasma physicist or the climatologist is working with classical theory which is basically understood. It is the theory’s reduction and application to complex problems that is at issue. Thus, the climatologist and plasma physicist frequently have a greater need for immediate feedback, for instance, to see if the evolution corresponds to preconceptions. Further, they typically spend more

* Early results, however, do show significant overlap for large granularity applications multitasking on the C90.
time developing code and less time running production than those in some other disciplines. Consequently, they perceive the need for an environment which allows the steering of simulations and the viewing of results in real-time.

“A computer environment which supports generating and storing enormous quantities of data, but lacks the powerful data navigation and visualization tools necessary to determine quickly and effectively whether the data should be archived or destroyed soon becomes dysfunctional. Interactive access to data with such tools should be supported at the highest level. Placing these concerns behind support for batch processing to fully utilize cycles can lead to a bankruptcy of the intellectual process.”

Clearly NERSC must seek to provide solutions sympathetic both to fire-and-forget batch production and to interactive or otherwise steered computing in the MP environment if it is to serve the scientific requirements of all its users.

Lessons Learned From the LLNL Experiment in Parallel Computing

Nielsen described what has since come to be known as the “Livermore Model.” Some of these concepts crystallized as a result of the experience at the Massively Parallel Computing Initiative (MPCI), a three-year experiment funded by the Director of LLNL and led by Eugene Brooks, to understand the requirements for efficient and productive use of parallel machines. Rather than repeat at length Nielsen’s arguments, we will summarize two areas on which we are concentrating at NERSC.

The first argues that a flexible space- and time-shared environment combined with the capability to dynamically partition the machine down to one computational processor is necessary. This environment would allow many users to use the machine simultaneously for (a) code development, (b) capacity production, (c) capability production, and (d) the transition from sequential to parallel applications. A dynamic, robust time-sharing scheduled environment is not currently standard on most parallel machines. As a partial solution, most modern parallel computers have supported the notion of space-sharing, which offers many users access to separate parts of the multiprocessor. Some vendors have implemented a time-sharing capability, as on the CM-5 and the Paragon.

The gang scheduled environment implemented at MPCI on the BBN TC-2000 by Gorda includes a scheduling mechanism which supports space-sharing and resource management. Time sharing is implemented on top of this. Beyond this, the users can specify the mode in which they intend to utilize the machine; for instance, they can specify “interactive” or “production” (batch), or even “benchmark.” How and when the time slices are allocated depends on the mode selected. In the figure on the following page, we see an example of a typical situation. The user has asked the “gangster” text-window driven tool to give him an instantaneous view of the processor environment. Each job is designated by a unique number or letter. In this figure only 59 of the 128 nodes are seen, and the “n” nodes that the user wishes to use for this time slice are visible. Clearly, if there are 1024 nodes instead of 128, this particular display would not be helpful; however, an intuitive generalization is possible. If the “gangster” tool were to be generalized to a 32 by 32 (or 64 by 32) rectangular grid in an X-window display (much like the Nielsen model diagram), each job could be assigned a particular color. The user could then locate his job at a glance and instantaneously obtain a picture of machine load.
LLNL and Cray Research, Inc. (CRI) share a CRADA to work together in the development of a space- and time-shared scheduler. It is hoped that collaborations of this nature, leveraging off the MPCI software development, will catalyze the development of flexible scheduled environments on all the parallel computers utilized by the LLNL and NERSC user base.

The second element of the "Livermore Model" most pertinent to NERSC relates to programming models. Each algorithm employed in an application code is best implemented utilizing a particular programming model (message passing, shared or data parallel); therefore, the programming environment should make as many models as possible available. In fact, a mixed programming environment is preferable.

A machine which has a remote reference capability built into the hardware, that is a machine which allows a processor to access the memory of another processor directly without disturbing the foreign CPU, allows efficient implementation of all three programming models. While this is not the only way to permit multiple programming models, a number of vendors of leading edge machines are offering a global address space effected in the hardware, or something very close to it. These include, for instance, CRI and Meiko, among others. The MPCI implemented both a message passing model and shared memory programming models, Parallel Fortran (and C) Preprocessors (PFP and PCP), which use a split-join programming technique. Currently, as part of a Livermore Directed Research and Development grant, Gorda and Warren are developing a version of High Performance Fortran (HPF) for remote reference capable computers. This code is called the Parallel Data Distribution Preprocessor (PDDP). Once HPF code is run through the preprocessor, PFP code is generated. NERSC is interacting with vendors who are developing machines with remote reference capability, and who are interested in using this code as a tool to understand possible bottlenecks and inefficiencies in implementations of HPF.

Some preliminary results utilizing PDDP are now available. The multigrid kernel of the Parallel NAS Benchmarks contains a 27-point operator. We have implemented a 3-point operator based on this 27-point operator. Initially we coded this operator in the PFP model, utilizing the BBN TC2000's interleaved shared memory facility to spread data accesses across the machine. This enabled the code to take advantage of the parallelism in the switch. Next, we coded this operator in PDDP, taking advantage of local memory speeds. The graph on the following page shows the execution time of each of these approaches on a 350,000 element vector. The first-cut implementation of the PDDP model produces code which is slightly faster than the PFP code. Additional work to reduce the overhead in memory placement algorithms will improve PDDP's performance.
The (Initial) Production Environment

Summarizing the preceding discussion, we see three general constraints in defining a workable production environment for a center such as NERSC. These are associated with the three phases in the life of a computational study: development, production, and analysis. The environment must

1. provide for the rapid development of efficient applications codes (development),
2. provide for the needs of both capacity and capability users (production), and
3. provide a balanced infrastructure to accommodate the data generation capabilities of the MP resource (analysis).

Rapid development of efficient applications codes (development)

To expedite development of applications, the user requires instant access to the resource. This functionality is achieved through a space- and time-shared scheduler controlling an interactive pool of processors. This implies hardware able to effect rapid I/O. In conjunction to this, two tools are needed: a sophisticated X-window based debugger, and a performance analysis tool. Other parallelization tools would be useful, but this discussion concentrates on minimal requirements for an initial production environment.

Beyond this, an effective mechanism for code migration onto the MPP should be enabled. One approach would be to migrate the application onto a single node of the MPP, and then parallelize modules one by one. This would then have the aspect of a tightly-coupled distributed calculation, with one node holding the remainder of the “old” sequential code and the remaining nodes...
running the parallelized modules (a similar procedure using the front-end for the sequential pieces could be utilized on the CRI T3D). This migratory approach represents a feature of the Nielsen model, and it requires that the individual nodes be both sufficiently fast and have sufficient memory to run most applications in their entirety at reasonable resolutions. For this to be a viable option, the MPP should be partitionable down to one compute element.

Since the applications base is varied, efficiency dictates that some combination of hardware and software should enable at least two of the three programming models as well as a mixed programming environment; it would be preferable if all three models were present, because in most cases, if one model were to be excluded, it would be the "shared" model, and this is probably the easiest to implement for the user, especially if the user has experience in multitasking on production CRAYs.

However, one glaring impediment to the generation of efficient codes now manifests itself, and this is the lack of a comprehensive set of linear algebra libraries to facilitate the choice of algorithms for applications. For example, fully implicit three dimensional discretizations could generate sparse matrices on the order of many millions. The solution of these systems on any parallel computer is beyond the domain of interest or expertise of most computational physicists. The development of parallel libraries has been inhibited by the diversity of architectures and programming languages creating a cybernetic Tower of Babel. Even with the development of common programming languages, architectural differences between machines would call for fine tuning. Implementations of the three levels of the BLAS would be a partial, interim solution. Then, existing sequential linear algebra packages could be linked with BLAS routines, providing a route to some parallelism. Linear algebra will remain a weak point in 1994 and beyond, and this represents an excellent area for HPCCP research.

Needs of capacity and capability users (production)

With a machine which offers (1) dynamic partitioning of processors, (2) CPUs that feature both high peak speeds (≥ 100 MegaFlops) with large memories (≥ 64 MegaBytes), and (3) high speed parallel I/O, implementation of a gang scheduler should provide for the legitimate needs of capacity as well as capability users. Rapid I/O is currently the most difficult of these requirements to effect. At the highest level, the machine would typically be divided into two partitions, a production (or batch) area and an interactive area. Capability jobs would typically dominate the batch partition. The interactive environment could be utilized for code development or steered computing, with individual jobs using small or moderate numbers of CPUs to minimize I/O costs associated with job swapping. Slice times in the interactive area would be much smaller than in the batch area. Capacity jobs could be directed at either partition (batch or interactive).

This represents a workable solution for capacity workloads. These applications could remain purely sequential, where they would run well on a fast CPU, or they could be rewritten to run in parallel. Numerous capacity jobs would not scale well to hundreds of processors. Nonetheless, many would benefit from parallelization if run on a smaller number of nodes. A fraction of the processors at any given time would be running sequential jobs. These could represent applications which are in the first phases of transition, as well as applications which are intended to remain sequential. The MP could then be viewed as an infinite source of virtual workstations (not that we are recommending carrying this view to extremes). The use of only 50 nodes in this manner could provide from 5 to 10 Gigaflops of workstation compute power.
A balanced infrastructure to accommodate the data generation capabilities of the MP resource (analysis)

Once the applications are running well, there remains a need to store and navigate intelligently through the data generated. To get a sense of the magnitude of the problem, a moderately resolved climate model simulation (requiring a 240-hour run time at a sustained 100 GigaFlops) would generate 23 MegaByte/sec for a total of 23 Terabytes of data. We present three of many options to address this problem.

(1) Expand and enhance current capabilities in storage

The current storage capacity at NERSC is about 12 Terabytes. One approach would be to expand capacity at the Center to allow the storage of a huge data set for a short period of time, long enough for a separate system controlled by a graphics engine to create a video. The original data set could then be destroyed. This might require the regeneration of the data at some later date. Regeneration of data is actually a good strategy in an environment in which the CPU to storage cost ratio is becoming ever smaller. A sensible long-term solution for storage might be the utilization of optical tape which has a single-reel capacity of 1 Terabyte, but has a very slow write speed. These tapes could represent the last tier of the storage hierarchy.

Storage and data retrieval are subjects of research within the HPCC Program. The National Storage Laboratory (NSL) at LLNL (housed at NERSC) is one such study. It represents a collaboration between ten vendors, lead by IBM, who are interested is assessing reasonable directions for the future. The requirements are for high storage capacity and high bandwidth throughout the hierarchy. This collaboration is open to other vendors and laboratories who bring appropriate technical contributions to the effort. The essential idea is that a very fast HIPPI switch routes data generated by an MPP or a C90 directly into a device, whether it be a solid state disk, a tape library, some other disk array or even a frame buffer. Thus, the data never goes directly through the funnel of a router, but rather the traffic is directed by IBM RISC storage access and management computers through a control network. A number of scientific programs are contributing pilot projects to this effort, including the climate effort at LLNL. This leads us to a second approach to the problem of assimilation of voluminous data.

(2) Real-time visualization and destruction of data

This strategy would feature a distributed calculation in which a high-end workstation generates the video as the calculation runs on the MPP. Data would be piped directly to the workstation, processed and then discarded. This is one of the strategies contemplated by the NSL. An interesting variation would be to allow the workstation to interact with the simulation and steer it. Alternatively, the rendering could be done in parallel on the MPP with the use of a workstation for ancillary tasks. For instance, this is the approach being taken by TMC with the CM-5 and will be used heavily at the ACL. There is a third possibility which is related to the first two but requires some additional planning by the programmer.

(3) Storing a generator set

Here the user determines the irreducible minimum of data (a generator set) which will allow him to reconstruct his problem. For a climate calculation, this might be the prognostic variables, and little more. These would be saved at appropriate intervals in the storage device as the MP program
marched through time. A postprocessor could be used to regenerate every detail of the calculation at the points in time at which the saves were made, allowing for the visualization of any aspect of the problem. In this mode, a long-running calculation would leave a trail of generator sets. If the time dependent interactions in a certain time interval are of interest, the MP computer could be used to run the problem through the relevant interval in time, at higher resolution if necessary. At this point, the strategy described in the previous paragraph could be utilized to generate real-time video.

How these and other strategies (including rendering in parallel on the MP itself) will be managed is the subject of investigation at the NSL, the Advanced Computing Laboratory, Argonne National Laboratory, and various other sites, all of which are highly motivated to provide at least some interim solutions by 1994. This problem demands an answer if a balanced, full-production environment is to evolve. There will be significant penalties to be paid if balance is not maintained as capacity is added. In the end, the community may have to adopt new policies and procedures which reflect the realities and costs of the new computing environment.

Summary and Conclusion

During the transition period, NERSC will have an advantage due to the presence of a C90 for production workloads. This represents a critical, but transitory advantage, since it will give users a choice about when to move over, and will remove some of the pressure that might otherwise be generated on the early MP production environment. Careful studies point out that the C90 will compete with or even dominate parallel computers during this transitional time frame. While parallel computers are rapidly improving, the conventional competition is not standing still. For instance, following the success of the C90, the 64 GigaFlops Triton should be formidable machine. By these measures, it has been argued that parallel machines are not yet ready for prime time. One could ask whether the Triton represents a viable alternative.

This paper will have served its purpose if it points to a plausible answer to this question. The Triton is being designed for the true mainstream, the corporate customer whose primary concern is the expeditious generation of numbers with an existing code. The NERSC user base, typical of most academic and laboratory research groups, represents a community which by its nature has to test the limits of available technology. That is where the leading edge computational results will be found.

In addition, the production environment, if it is to evolve, cannot do so without the participation of the research community mainstream. As explained earlier, the vital pieces of the production environment are the application codes and the human intelligence needed to create these applications. The development of a single aspect of the full production environment cannot proceed in isolation. The presence of the early production environment makes possible (and requires) the movement of MP computing to the research community mainstream. It will indeed be possible to offer a code development and early production environment by 1994. This will require insight on the part of NERSC in determining and requesting the indispensable features of the environment. This will also require an understanding on the part of the users that the inevitable uncovering of shortcomings in the system is their opportunity to determine its evolution. The transition will be characterized by a highly interdependent, indeed symbiotic, relationship between the environment and the users. This period will represent a great adventure for those who wish to take up the challenge.
Work at Lawrence Livermore National Laboratory performed under the auspices of the U.S. Department of Energy under contract number W-7405-ENG-48.
References


5. G. D. Kerbel, NEPSC, private communication.


END

DATE FILMED

11/16/93