BNL--34185

DE84 010513

TRACKING AND COMPUTING

J. Niederer

Brookhaven National Laboratory, Upton, New York 11973

BNL-34185

Summary

This note outlines several ways in which large scale simulation computing and programming support may be provided to the SSC design community. One aspect of the problem is getting supercomputer power without the high cost and long lead times of large scale institutional computing. Another aspect is the blending of modern programming practices with more conventional accelerator design programs in ways that do not also swamp designers with the details of complicated computer technology.

Introduction

Tracking of particles by computer is now widely used to explore slowly growing instabilities which may be driven by higher order nonlinearities in guide fields, and by various forms of weak interactions among beams and their electrical environments. Many of the large number of possible design options and parameters available for the SSC are possible candidates for being confirmed, or even optimized, through the use of potentially very long simulation runs. The adding of further complexity to the various tracking models is now often discouraged by computing cost tradeoffs. Moreover, the detailed analyses required may well demand more computing power and real memory than is provided by the larger computers presently on the scene, irrespective of costs. Besides issues of cost and available computing power, the usefulness of tracking methods is also affected by the ease with which they are applied and modified, and the overall time it takes to adapt them to changing parameters.

Three broad areas of support may be described. The first involves the supply of economical, massive computing power in ways that are available to the several SSC design groups. The second includes reformulating tracking procedures so that the higher order instabilities are emphasized. By transforming away lower orders, in principle very long term high order effects may be studied on computers with single precision, and hence with greater speed and lower cost. The third area includes common library services, such as flexible data base operators and structures, and common data entry and display services. Each approach may be pursued in parallel with the others, and each will contribute to making the simulation methods more practical and powerful, and also reduce computing inconveniences to accelerator designers.

Computing Power for Tracking

Three main options are available which are likely to be used in combination. These are expanded use of present large computers, a shift to more economical specialized computing engines, such as array processors, and lastly, highly specialized hardware, optimized for the most time consuming of the tracking algorithms. Fischler and Nash[1] discussed these options in detail in this Workshop, emphasizing the computer technology involved in the latter. There is a progression from highest cost to lowest cost in these options, a progresaion from easiest to hardest to use by an accelerator (not computer) designer, one from most to least flexible, and one from least to most powerful for specific types of analysis.

Tracking programs with nonlinear terms normally spend over 90% of their time evaluating a complex polynomial form of local impulses of kicks. These kicks are typically expressed as a complex polynomial of field coefficients B_n and the transverse coordinates of a track, $r = x + i y$:

$$K = \sum_n B_n r^n$$

Hence, schemes for speeding up tracking concentrate on handling this series. Array processors, tight inner loop coding, special series coprocessors, and table lookups are ways to help, if the algorithms remain stable. However, their application has to be thought out carefully, as a strategy which may be efficient for computing may limit the range of simulation that can be done. For example, table lookups involve tradeoffs of accuracy of tables versus total number of nonlinear impulses around the ring.

Unfortunately, the development of specialized computer hardware and software has invariably been far slower than the time to deliver accelerator design results by conventional means. For the newer methods to be useful, times for specialized developments have to be brought more in line with the time demands for results. If, indeed, this delay problem can be mastered, designers will also need constant collaboration with highly skilled computer people to exploit these support tools.

Restating the Tracking Problem

Currently used tracking programs all compute the effects of discrete deflecting elements upon a circulating particle. In all, a lattice is approximated as a set of alternating linear and nonlinear elements. There are two basic ways that these effects are evaluated. In the MARYLIE approach[2], transport expressions for the discrete elements of an appreciable section of the orbit are combined algebraically into a map. The resulting section map expression is evaluated, and applied to advance the track orbit through the section. In most other programs, the orbit is evaluated piecewise at every element. In practice, the elegance of the Lie method does not extend easily to high orders of multipole influence. Hopefully, a combination of the thorough MARYLIE algorithms for nominally linear groups of elements together with the discretely evaluated impulse treatments will be developed, yielding the advantages of both approaches.

In principle, the higher orders are levels of perturbed motion superposed upon the basic linear lattice motion. Computationally, the impulses, even when multiplied by the distance to the next focussing or perturbing element, appear in the fifth or sixth significant figure of the track coordinates. This all but forces use of double precision on computers with 32-bit word size; the penalty in speed for double precision is about a factor of five. Ideally, these higher order calculations should be done in an appropriate perturbation representation, which places the effects being studied into the most significant portions of the working coordinates. Finding a suitable scheme will permit meaningful higher order computations with 32-bit array processors and the more specialized microprocessor hardware.

Array Processors For Tracking

The relative merits of array processors (AP) can be demonstrated for the demanding inner loops of the

nonlinear impulse codes. Conventionally, the complex series is arranged in the form:

$K = B_1 + r\,(B_2 + r\,(B_3 + .. r\,(B_8 + rB_{10})\,..)\,)$

written here for ten orders of multipoles. This reduces to four additions and four multiplications per order. As implemented in LILA[3] (Patricia) on the BNL CDC7600, these are executed at an effective rate of 8 MFLOPS (Million Floated Operations Per Second), corresponding to about 10 μsec to compute one impulse. This assembly code is designed to keep all of the parallel adders and multipliers busy; new instructions are issued as soon as an arithmetic unit is freed, up to a maximum of one every 30 μsec. However, along with each eight floating point operations per order of multipole, approximately thirty other instructions must be issued for loading or storing operands, etc. With these and other overheads, the LILA program covers about 60 turns per second for a CBA sized lattice with 700 impulse sites. The DESY RACETRACK program[4] used on the HERA lattice shows comparable statistics on a large IBM computer.

Newer array processors appear to be significantly faster for this kind of problem because they can issue operands to all arithmetic units in the same cycle, and they use parallel service processors that do not steal instruction issue cycles from the arithmetic units. In one example, a theoretical limit of 100 MFLOPS is claimed for optimal code that issues operands to each of two add and two multiply units each 40 μsec cycle. A paper exercise suggests that this particular AP will do specialized serial tracking and impulse algorithms over four times faster than the 7600, effectively yielding over 30 MFLOPS. These numbers are attractive enough to encourage more careful coding with lattice data streams organized to take advantage of this type of computing engine.

Other features needed for large scale tracking calculations appear to be well considered in the newer AP designs. Memory is adequate for large lattices, and expandable. Data flow rates, and cache and buffering can keep the arithmetic units supplied with nonlinear series coefficients. There are no obvious conflicts between the tracking problem organization and the opportunities for using the optimizing features of the AP. The apparent cost is about $10K per useful MFLOPS, an order of magnitude under conventional large scale computer figures.

The picture is less rosy with software, but workable. To achieve dramatic computing speeds requires careful micro coding of up to about 1000 inner loop instructions, and reorganization of the problem data flow to suit the processor. Thus the code executed may not resemble the original statement of the lattice physics particularly well. As supporting software has always been a problem with specialized processors, an AP applications expert should be included in any group which elects to use these devices for simulation studies.

Common Data Base Program Tools

Accelerator design lends itself well to industry-wide common program tools for designers. These emphasize ease of data entry, familiar vocabularies, interactive features, and hidden programs which carry out the boring details of assembling lattices and propagating design changes through them. In tracking programs most of the code is concerned with accepting and checking lattice configuration data, assembling it into forms such as transport matrices for tracking, and displaying various stages and results of the computations.

The MAD group of programs at CERN[5] is a modern example which offers excellent, well documented input features and couples related programs with compatible intermediate formats. The BNL LILA work has concentrated on more formal relational data base methods, developing basic operators and structures that can handle a wide variety of programs, lattices and orbit features. These and similar prototype work of other groups[6,7] have shown the merits of beginning with data organization, and working the physics parts off data structures which can easily be common to all such programs.

The basic idea is that a designer makes a list of classes of objects, such as bends, quads, etc., and names a list of properties needed to describe each class, in a sense defining a vocabulary for the lattice. Then each element of the lattice is described with simple statements that relate it to the class definition; "Jones is a BEND with Strength of 20. and Length of 1.", etc. Hundreds or thousands of such statements are dropped into a large pot (memory) where relational data base routines sort them out, building up data structures based upon the class definitions. The designer works only with the names of objects; he is spared the tedium of dealing with thousands of items and their properties. Given the basic data, a lattice now amounts to a list of names of elements; the data for each element tells a physics application program what should be done with it. Most lattice changes reduce to simple editing within the group of input statements.

This type of relational data base could be provided for the accelerator community in quite general form in perhaps a year. It depends upon liberal use of computer memory, and needs clean bridges to the Fortran written physics parts of tracking programs. Most of the necessary specification and user documentation, and much sample code, can be drawn from the MAD, LILA, COMFORT[7] and other prototypes. Studies which make use of specialized computing devices would similarly begin with the common data structure, reorganizing parts of it into forms optimized for tracking as an initialization step in a host computer.

Dedicated Computer Workstations For Tracking Exercises

Design studies at BNL for the CBA[8] incorporated major simulation capabilities into the control system. We intended to provide the accelerator physics features of the design programs for guiding beam adjustments directly at the consoles. This has given a close view of the design and simulation programs and their computing needs, and also a practical reason to examine the technology, costs, and tradeoffs involved in implementing them on line. The CBA designs foresaw that commercial developments would produce capable 32-bit microprocessor based computers with essentially unlimited memory so cheaply that software could be based from the start upon large memory data base procedures. Furthermore, they anticipated that raw computing power in the form of computing engines would be available at costs low enough for routine consideration in the control system. Very similar conclusions appear in the LEP controls designs.[9] Indeed, modern 32-bit workstations with superb graphics, over 1 MIPS performance, very large memories, and extensive software of large computer system quality are now on the market. These are being obtained at BNL to support our efforts to introduce interactive features and data base facilities to accelerator simulation programs, together with an immediate application in improving the AGS control system. They will also be tried as replacement consoles for the AGS. Shering[10] has discussed a similar role as possible LEP consoles.

It may be possible to test the usefulness of these ideas for the tracking problem quite soon. At BNL, an array processor is being considered for central computing, as an attractive, general, in-hand solution for a number of computing intensive problems. Our CBA experience suggests that the combination of a host workstation and array processor may be an excellent tool for dedicated simulation use by an SSC design group. A well-equipped workstation with large disk costs about $100K, and additional simpler units ($15-25K) join easily into a common network. Array processors are in the $250K range. The data bases, program libraries and compilers, and specialized compilers and drivers for the AP or computing engines reside in the workstation. Software closely resembles that of the larger minicomputers, and the Fortrans are very compatible with both IBM and DEC ones. As they evolve in time, more specialized, privately developed computing engines and coprocessors can also be joined readily to the standard busses and interfaces of the workstations. The workstations and their various appendages appear as rather elaborate graphics terminals to the user.

References

1. Fischler, Mark and Nash, Thomas, "Computing Tools for Accelerator Design Calculations", Dec., 1983, This Workshop.

2. Dragt, A.J., "Lectures on Nonlinear Orbit Dynamics", Physics of High Energy Accelerators - Fermilab Summer School, 198., AIP Conference Proceedings #87, 1981.

3. Niederer, J. and Morris, B., Proc. Workshop on Accelerator Orbit and Particle Tracking Programs, Brookhaven National Laboratory, May, 1982, (BNL 31761, New York, 1982), p.26.

4. Wrulich, A., ibid, p.277.

   Wrulich, A., This Workshop.

5. Iselin, F.C., "The MAD Program", 12th International Conference on High Energy Accelerators, August, 1983, Fermilab. (CERN-LEP/TH/ 83-30).

   Iselin, F.C., "The MAD Program: Reference Manual", CERN, October 20, 1983, (Geneva, Switzerland).

6. v.Egan-Krieger, G., Klotz, W.D., and Maier, R., "The Minicomputer Network for Control of the Dedicated Synchrotron Storage Ring BESY", Europhysics Conference: Computing in Accelerator Design and Operation, Berlin, W.Germany, September, 1983.

7. Woodley, M.D., Lee, M.J., Jaeger, J., and King, A.S., IEEE Trans. Nucl. Sci., NS-30, August, 1982.

   Lee, M.J., Sheppard, J.C., Sullenberger, M., and Woodley, M.D., "Models and Simulations", Europhysics Conference: Computing in Accelerator Design and Operation, Berlin, W.Germany, September, 1983 (SLAC - Pub. 3214, Sept., 1983).

8. The Colliding Beam Accelerator, Design Proposal, Brookhaven National Laboratory, April, 1983.

9. Crowley-Milling, M.C., "The Control System for LEP", CERN-LEP/DI/ 83/20, 1983.

10. Shering, G., "Consoles and Displays for Accelerator Operation", Europhysics Conference: Computing in Accelerator Design and Operation, Berlin, W.Germany, Sept., 1983.