APPLICATIONS OF PARALLEL PROCESSING TO ASTRODYNAMICS

S. COFFEY, L. HEALY AND H. NEAL[1]
Naval Research Laboratory
Washington DC, USA

Abstract. Parallel processing is being used to improve the catalog of earth orbiting satellites and for problems associated with the catalog. Initial efforts centered around using SIMD parallel processors to perform debris conjunction analysis and satellite dynamics studies. More recently, the availability of cheap supercomputing processors and parallel processing software such as PVM have enabled the reutilization of existing astrodynamics software in distributed parallel processing environments. Computations once taking many days with traditional mainframes are now being performed in only a few hours. Efforts underway for the US Naval Space Command include conjunction prediction, uncorrelated target processing and a new space object catalog based on orbit determination and prediction with special perturbations methods.

1. Background

A decade ago parallel processing became available commercially from companies like Thinking Machines, MASPAR, INTEL, IBM and others. Since then many hardware and software environments have been available. In this paper we present an overview of problems to which we have applied parallel processing. The applications include the critical inclination problem in celestial mechanics and operational problems encountered by the US Naval Space Command (NSC), one of the primary US tracking organizations. The techniques we employed have varied: we have used SIMD (Single Instruction Multiple Data) massively parallel computers, and distributed parallel systems built on the PVM (Parallel Virtual Machine) environment.

[1]GRC International, Colorado Springs, currently located at Naval Research Laboratory

Celestial Mechanics and Dynamical Astronomy **66**: 61–70, 1997.

There are several ways that parallel processing can benefit the user:

- improve turnaround time,
- more efficient utilization of hardware,
- reuse of existing software in a new computing environment.

In the applications we have built, many of these benefits have been realized.

2. The Critical Inclination Problem

The Naval Research Laboratory (NRL) obtained a parallel processing computer, the Connection Machine 2 (CM-2) in 1987. Our first effort with this computer was to study the critical inclination problem for the theory of an artificial satellite. After averaging, the phase space for this problem is a 2 dimensional sphere which can be visualized graphically. We set out to develop a method for using the new parallel processor in our investigations. We started with a serial program called CCCP (Creative Color Contour Program) developed at NIST by Jonathan Aronson working in collaboration with Dr. André Deprit. This program, written for a LISP Machine, allowed one to display contours of a Hamiltonian by assigning colors to the Hamiltonian values. This eliminated the need for tedious numerical integrations, and produced a more complete picture than previously possible.

The Hamiltonian for the artificial satellite problem was constructed by averaging the short period terms when the perturbations were restricted to the zonal harmonics. We give here the Hamiltonian through J_4,

$$\begin{aligned}\mathcal{H} = {} & (\tfrac{G}{p})^2(\tfrac{\alpha}{p})^2\eta^3\Big(\tfrac{3}{4}s^2 - \tfrac{1}{2}\Big) \\ & + J_2(\tfrac{G}{p})^2\Big((\tfrac{\alpha}{p})^4\Big(\xi_2^2\eta\Big(\tfrac{45}{32}s^2 - \tfrac{21}{16}\Big) + \eta^5\Big(\tfrac{75}{128}s^4 - \tfrac{27}{32}s^2 + \tfrac{3}{16}\Big) \\ & \qquad + \eta^4\Big(-\tfrac{27}{32}s^4 + \tfrac{9}{8}s^2 - \tfrac{3}{8}\Big) - \eta^3\Big(\tfrac{195}{128}s^4 - \tfrac{81}{32}s^2 + \tfrac{15}{16}\Big)\Big) \\ & \quad + \tfrac{J_3}{J_2^2}(\tfrac{\alpha}{p})^3\xi_2\eta^2\Big(\tfrac{15}{8}s^2 - \tfrac{3}{2}\Big) + \tfrac{J_4}{J_2^2}(\tfrac{\alpha}{p})^4\Big(\xi_2^2\eta\Big(\tfrac{105}{32}s^2 - \tfrac{45}{16}\Big) \\ & \quad + \eta^5\Big(-\tfrac{105}{128}s^4 + \tfrac{45}{32}s^2 - \tfrac{9}{16}\Big) + \eta^3\Big(\tfrac{315}{128}s^4 - \tfrac{105}{32}s^2 + \tfrac{15}{16}\Big)\Big)\Big).\end{aligned}$$

The variables in the Hamiltonian are given by

$$\xi_1 = \eta e \cos g, \quad \xi_2 = \eta e \sin g, \quad \xi_3 = \eta^2 - \frac{1}{2}(1 + H^2/L^2).$$

In these coordinates,

$$\xi_1^2 + \xi_2^2 + \xi_3^2 = \frac{1}{4}(1 - H^2/L^2)^2. \tag{1}$$

The remainder of the variables are e, the eccentricity, H the angular momentum, L the conjugate to the mean anomaly, $\eta = \sqrt{1 - e^2}$, and $s = \sin(I)$.

https://doi.org/10.1017/S0252921100046376 Published online by Cambridge University Press

The Hamiltonian has one degree of freedom after averaging, with two integrals, H and L. Thus, interesting information was to be had by studying the evolution of the phase space under the control of the integrals.

Using CCCP as a guide, we wrote a new program for the CM-2, although with new algorithms to enhance the details of the phase space. Conceptually, the idea is to assign different processors, from a pool of 64K processors, to the coordinates of a grid placed on the projection of the phase space onto a plane, Figure 1. The parallel processor then computes simultaneously the Hamiltonian for all points in the grid. The Hamiltonian values are sorted and assigned colors which are used to paint the phase space. The initial idea was to construct a rectangular grid on the projection of the phase space. Thus, we first select an orientation then apply the coloring algorithms to paint the resulting projection. The full details behind this technique are provided in Healy and Deprit (1991). This technique provided a lot of information quickly that allowed us to gain an appreciation of the changing phase space. From these observations we drew conclusions that helped direct subsequent analytic investigations. The results of the research can be found in Coffey *et al.* (1994).

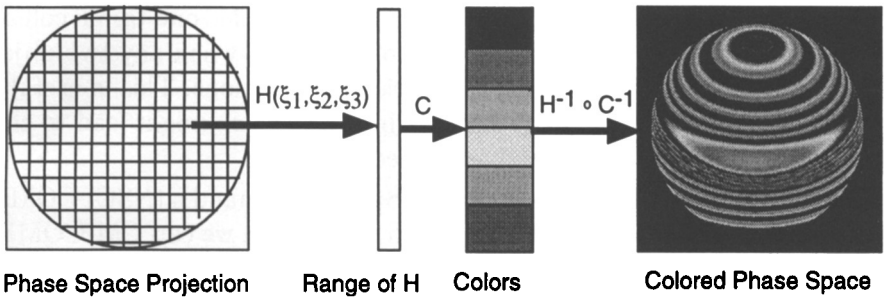

Figure 1. Coloring a Phase Space.

Since the phase space is a sphere, it is important to see the image from different orientations. However, the CM-2 is not a good graphics engine. Using an icosahedral (nearly uniform) grid for the sphere (Baumgardner and Frederickson, 1985) we can compute the Hamiltonian on the sphere rather than on the projection of the sphere. This enables us to bring back the true 3 dimensional phase space for display on our Silicon Graphics workstations. This provides better visualization of the three dimensional images. We frequently use 40962 vertices for the true 3-D images.

A SIMD computer, like the CM-2, was a good choice for this problem. Here we were building a new computer program and were free to select the hardware best suited to the problem. For many subsequent problems we did not have this luxury in that we often have to reuse existing software, which

carries with it the need to conform to certain hardware. SIMD stands for Single Instruction Multiple Data, which means that at each execution cycle every processor executes the same instruction. For the phase space problem we needed, at each grid point, to evaluate the Hamiltonian. The algebra is the same regardless of the coordinate. Only the coordinates which are the data are different. This program could therefore be written with huge internal arrays to hold the Hamiltonian values. Each array element would be automatically mapped to a processor by the Connection Machine system. This computer system provides special hardware for efficiently broadcasting instructions to the processors and returning values to the control processor which is the front end computer.

3. Satellite Collision Prediction

Another problem that we applied our CM-2 computer to was COMBO, the Calculation of Miss Distance Between Objects in space. We wanted to determine when objects in space come close to each other. The Air Force and Naval Space Commands have had for a number of years, programs for computing the distances between a single object and the set of known objects in the space object catalog. We call this the one-to-all collision problem. We wanted to generalize this to determine when any two objects come close together, an all-to-all comparison of positions. This is useful for determining the probability of collisions in space which might lead to chain reactions of fragmentations of space objects.

Although we had access to the Naval Space Command's (NSC) COMBO software, we elected to build from scratch, a program we call CM-COMBO. The hardware of choice was again the Connection Machine. At first we used the CM-2, but, later we converted to the Connection Machine 5E (CM-5). This computer runs similar to the CM-2, but is quite different in hardware. Its processors are 64 bit versions of the Vector Sparc Chip. There are fewer of them, 256 processors in our computer, with 16K possible. The technique on this computer is to allow each processor to act like a large set of virtual processors. That is, each processor cycles through arrays much like a serial computer does with a do-loop. This is transparent to the user, the programming being almost identical to that on the CM-2.

The concept behind CM-COMBO was to assign individual processors to the objects in space. Each processor starts with the initial conditions, an element set, for its object. It applies a propagator to the elements to compute the position for all objects. at the next time step. We generally use the propagator PPT2, which is what NSC uses to construct the elements for the Navy's catalog. PPT2 is based on the analytic theory of Brouwer (1959). Since the propagator is analytic the same instruction set is used to

Figure 2. Conjunctions of Debris for catalog of 3 June, 1996. Catalog size of 8919 objects. Conjunction distance of 5 km.

compute the position of every object. Thus, again we see the benefit of the SIMD computer. A fairly complex method for determining close approaches was developed, the details are available in (Healy, 1995). Although COMBO prints a report on the conjunctions, the most flashy output is a picture like that in Figure 2. This particular figure shows the instantaneous locations of the objects in the catalog for December, 1995. The + marks indicate two or more objects less than 5 km apart.

NSC runs COMBO remotely from an account on the CM-5. The element sets are automatically deposited in the account. NSC has used the program to confirm conjunctions with the shuttle which they obtain with their own software. They also run the program to obtain reports of coming conjunctions that might be photographed by ground based telescopes. One such event of TIROS 10 and NOAA 10 was photographed on May 21, 1993. COMBO computed a miss distance of 2.1 km. This is the first significant use of external computing resources for operations at NSC.

A SIMD computer is a natural choice for this problem because of the use of an analytic propagator. However, if for increased accuracy, a numerical integrator were substituted then the efficiency of the SIMD computer would suffer because the huge variation in the orbits represented in the catalog would require substantially different step sizes for the integrations. Thus the computer would run at the speed of the slowest integration.

SIMD computers are dropping in popularity and likely will be less of a factor for parallel processing in the future. As a result Dr. Healy recently ported most of COMBO to the PVM (Geist *et al.*, 1994) environment. PVM allows a group of heterogeneous computers to operate as a parallel computer. Unlike the SIMD method of broadcasting instructions, information in

a PVM environment is passed by a message passing system. PVM proved an adequate system for this version of COMBO.

4. Uncorrelated Target Processing

US Space Command, in tracking space objects, frequently obtains observations that cannot readily be correlated to a known object. Often the observations come from an unknown object or from an object previously cataloged but later lost. Most often these observations come from fragments produced by explosions of rocket bodies and satellites. These observations, called uncorrelated targets (UCTs), present a serious problem for US Space Command. The Naval Space Command radar fence that stretches across the southwest United States is the largest single detector of UCTs. The fence receives radar returns for almost any object above 30 degrees of inclination.

NSC developed a program called SAD (Search and Determine) to correlate UCTs which are kept for 60 days. The essence of the algorithm, when it is used for UCT processing, is to form every possible pair of observations from the UCT file and then determine what orbits could contain these two observations. For UCT processing the observation set is usually made up of earth-centered X-Y-Z observations. These observations are created by triangulating direction cosines from the NSC fence to create one X-Y-Z observation per orbital pass.

SAD is also used for processing fragmentation events. Here, the blast point of the fragmentation is used as the first observation in every pair. In this case the number of pairs processed decreases dramatically with a corresponding decrease in computer time.

NSC has never been able, with their 1970-vintage Cyber computers, to process more than 5 days of observations, mainly because the computer time would take several days. Moreover, recent events at NSC eliminated any possibility of running the program. We undertook to rewrite this program for parallel processing (Coffey *et al.*, 1996) with three objectives:

- reestablish the use of the SAD for UCTs,
- provide rapid turnaround performance,
- enable the processing of the full 60 day sets of observations.

This is an instance where we reused most of the existing program. We rewrote the program to run on top of PVM. PVM links distributed processors together in a parallel processing environment. Each processor receives data, completes an assigned task and returns results to one or more computers. There are a number of nested loops in the original SAD program, starting with a loop through the pairs of objects called starter pairs, a loop over the direction of the orbits constructed and a loop on the possible solutions to Lambert's problem. Inside the starter pair loop there is a

refinement step to include associated observations. We parallelized on the outermost starter pair loop. A master/slave design was implemented. The master program loops over all possible choices for the first object in the starter pair. This object is called a starter point. Each starter point is sent to one of the slave processors along with the complete observation set. The slave combines the starter point with all choices for the second object in the starter pair and begins forming orbits. This code was unchanged from the serial program. The master starts up the slave programs, one for each node in our "virtual machine." The virtual machine is the collection of processors on which the program executes. The master can auto-detect the number of nodes available eliminating the need to specify how many nodes to use. Because the master program does so little work, a slave program is also initiated on the same computer.

The code internal to the starter pair loop was formed as a standalone program for the slave processors. The slave processors return the orbits found, providing a distinction for orbits presenting a high confidence of being a real object. These orbits are called superior orbits. The greatest dilemma in designing the parallel version of SAD was how to handle superior orbits. In the serial version of SAD, the discovery of a superior orbit affects the processing of all succeeding starter points because the observations linked to the superior orbit are removed from the observation set. In the parallel program, several starter points are processed concurrently, so when a superior orbit corresponding to a particular starter point is found in one slave program some of the observations supporting that superior orbit may have been already processed as starter points by other processors. In order to mimic the serial version of SAD exactly, all starter points after the one for which a superior orbit was found would have to be reprocessed with the adjusted observations set. This would require greater complexity in the code to take care of the necessary bookkeeping, and the re-processing could severely diminish the efficiency of the parallel code, especially when a large number of nodes are present in our virtual machine.

We have some leeway in handling superior orbits. However, we still wish to remove the observations associated with superior orbits from the observation set. Leaving them in can result in extra orbits in the output. Our solution is to update the observation set on the master and all the slaves as soon as possible after it changes. The slave program in which the superior orbit is found returns the orbit and its associated observations which are removed immediately from the observation set. The updated observation set is then broadcast to all the slave nodes. The slave nodes will not receive this set, however, until they finish processing the starter point on which they are working. So, the change on all the slaves except the one with the superior orbit will be effective with the next starter point

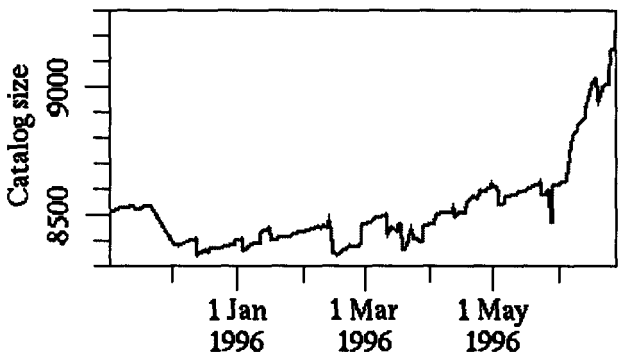

Figure 3. Increase in the Space Object Catalog.

they process. The starter points that are being processed or have already been processed are not re-evaluated with the updated observation set.

One implication of updating the observation set in this manner is that if Parallel SAD is run multiple times with the same observation set, the number of orbits and their solutions will vary from run to run. This occurs because the starter points that have been processed when the observation set is updated will be different on each run, due to differences in the operating conditions on the nodes in the virtual machine. This behavior makes it very difficult to test and debug the software, but it is acceptable from the analyst's point of view: any version of the output set will meet his needs. The program usually produces more orbits than the analyst can use. But it generally does not miss real orbits.

Parallel SAD was installed at the Maui High Performance Computing Center's IBM SP2 parallel computer located in Hawaii, where it became operational in May, 1995. Since then it has been used on average every two weeks by the operations group at NSC. Programs are submitted to the SP2 through batch queues, the largest allowing for 128 processors. The turnaround for a batched job can be as much as 24 hours. However, that is substantially less than it would take on a single workstation. The normal run time for 60 days of data is two to four hours on 128 processors. The programming time was approximately 3 man months. This is the second time that NSC has made significant use of external resources for operations.

Recently we installed the program at NSC on 14 little used workstations from their Finance Department. The installation took a day for PVM and Parallel SAD and 3 days to check out the program. However, interestingly, previous to installation it took 30 days to obtain approval to install the system. The normal execution takes about 16 hours on these computers.

On 3 June 1996 object 23106, a Pegasus rocket body launched in 1994, exploded in space. This has resulted in 539 fragments being cataloged,

most of which will stay in orbit for hundreds of years. Parallel SAD played a significant role in cataloging these objects. As can been seen in Figure 3 the catalog now contains over 9200 objects. We can attribute significant credit for the growth in the catalog to the use of Parallel SAD.

From our experience with Parallel SAD, we draw the following conclusions about parallel processing.

- Many existing programs can be parallelized at little cost,
- Dedicated computers are not necessary for parallel processing,
- Distributed Parallel Processing is mature enough for operational use,
- Networking can provide external parallel processing resources for operational use,
- Parallel processing provides a means to run jobs that would be prohibitively long on a single serial machine.

5. Catalog Maintenance with Special Perturbations

Since the beginning of the Space Age, US Space Command has used analytic propagators to maintain their catalog. Recently we undertook a task to develop a catalog using special perturbations, that is, with a numerical integrator. The reason is that numerical integrators can provide more accuracy through the use of better force models. Whether the radar data is sufficient to support the better accuracy needs to be determined.

Since this will require substantial computer time, we again are using parallel processing. Again we seek to adapt an existing program to run on top of PVM. The integrator itself is a variation of a Gauss-Jackson 8th order integrator already in use by US Space Command.

The concept here is fairly straight forward. The parallel system is written as a master/slave arrangement. The master computer organizes the I/O, starts up the slave nodes and sends them the initial state vector. The slave then integrates, differentially corrects the orbit state and sends it back to the master for cataloging.

This task illustrates the most primitive form of parallel processing. There is no communication between the slave processors, only between the master and the slaves, and then only for data transfer. The slaves are free to update their state vector in their own time. Once they complete one satellite, they return their state vector and receive a new one. Synchronization is quite simple since the master controls everything.

6. Conclusions

One method of parallel processing we have not found useful is what is referred to as control decomposition where dissimilar parts of a program are sent to different processors for execution. In this form of parallelization

great care must be exercised to maintain a high level of synchronization and utilization of the slave processors. We have have not found this type of parallelization useful for our applications.

This should not be confused with the method used on SAD where a core module was extracted and duplicated for the slave computers. The slaves performed a large task to completion on each data set from the master.

For each of our applications, the data for the slave nodes has an identical structure and the programs on the slave nodes is always the same. In our SIMD application the synchronization was at the instruction level with each processor receiving a copy of an instruction to execute on the data. In the PVM applications the synchronization was more coarse, being at the program level. Here each processor received a complete yet identical program to execute on data possessing an identical structure. Thus, there is a great deal of similarity in our applications whether SIMD or distributed parallel processing under PVM. For the PVM applications the flexibility of processors completing tasks at different times produces better efficiency for problems that cannot be precisely synchronized.

The environment, PVM, is available for most computers. As indicated, we have made extensive use of it and have found it to be very robust. PVM has spawned a standardized language called MPI which is now available.

Acknowledgements. Many of the projects reported on here were supported by Naval Space Command under the direction of Drs. Steve Knowles and Paul Schumacher. Thanks to all members of NRL Code 8233 for their efforts in making these projects successful. Special thanks to Dr. André Deprit who strongly urged us to use parallel processors when machines first became available.

Authors are listed in alphabetical order.

References

Healy, L. and Deprit, E.: 1991, "Paint by number: Uncovering phase flows of an integrable dynamical system," *Computers in Physics* Sep-Oct, 491–496.

Coffey, S.L., Deprit, A., and Deprit, E.: 1994, "Frozen orbits for satellites close to an Earth-like planet", *Celest. Mech. & Dyn. Astron.* **59**, 37-72.

Baumgardner, J.R. and Frederickson, P.O.: 1985, "Icosahedral discretization of the two-sphere", *Siam J. of Numerical Analysis* **22**, 1107-1115.

Healy, L.: 1995, "Close conjunction detection on parallel computer", *J. Guid. Control Dyn.* **18**, 824-829.

Brouwer, D.: 1959, "Solution of the problem of artificial satellite theory without drag", *Astron. J.* **64**, 378–397.

Geist, A.,: 1994, *PVM—Parallel Virtual Machine—A Users' Guide and Tutorial for Network Parallel Computing*, MIT Press, Cambridge, MA.

Coffey, S.L., Jenkins, E., Neal, H.L., and Reynolds, H.: 1996, "Parallel processing of uncorrelated observations into satellite orbits", *AAS/AIAA Astrodynamics Specialist Conference*, Austin, Paper 96–146.