12
4-10-95 JS①

UCRL-ID-119803

Massively Parallel Fast Elliptic Equation Solver
for Three Dimensional Hydrodynamics and Relativity

Patricia L. Sholl
James R. Wilson
Grant J. Mathews
John H. Avila

Lawrence Livermore National Laboratory
Livermore, CA 94551

January 1995

This is an informal report intended primarily for internal or limited external distribution. The opinions and conclusions stated are those of the author and may or may not be those of the Laboratory.

Work performed under the auspices of the Department of Energy by the Lawrence Livermore National Laboratory under Contract W-7405-Eng-48.

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

**Massively Parallel Fast Elliptic Equation Solver
for Three Dimensional Hydrodynamics and Relativity**

Patricia L. Sholl
James R. Wilson
Grant J. Mathews
John H. Avila

U. S. Department of Energy

Field Work Proposal

**Massively Parallel Fast Elliptic Equation Solver
for Three Dimensional Hydrodynamics and Relativity**

Principal Investigator: Patricia L. Sholl

Systems Research Group/ ERD

Lawrence Livermore National Laboratory

Co-Investigators:

James R. Wilson

Physics and Space Technology Program/ V-Division

Lawrence Livermore National Laboratory

Grant J. Mathews

Physics and Space Technology Program/ V-Division

Lawrence Livermore National Laboratory

John H. Avila

Mathematics and Computer Science Department

San Jose State University

ABSTRACT

Through the work proposed in this document we expect to advance the forefront of large scale computational efforts on massively parallel distributed-memory multiprocessors. We will develop tools for effective conversion to a parallel implementation of sequential numerical methods used to solve large systems of partial differential equations. The research supported by this work will involve conversion of a program which does state of the art modeling of multi-dimensional hydrodynamics, general relativity and particle transport in energetic astrophysical environments. The proposed parallel algorithm development, particularly the study and development of fast elliptic equation solvers, could significantly benefit this program and other applications involving solutions to systems of differential equations.

We shall (1) develop a *data communication manager* for distributed memory computers as an aid in program conversions to a parallel environment and implement it in the three dimensional relativistic hydrodynamics program discussed below; (2) develop a *concurrent system/concurrent subgrid multigrid method*. Currently, five systems are approximated sequentially using multigrid successive overrelaxation. Results from an iteration cycle of one multigrid system are used in following multigrid systems iterations. We shall develop a multigrid algorithm for simultaneous computation of the sets of equations. In addition,

we shall implement a method for concurrent processing of the subgrids in each of the multigrid computations. The conditions for convergence of the method will be examined. We'll compare this technique to other parallel multigrid techniques, such as distributed data/sequential subgrids and the *Parallel Superconvergent Multigrid (PSMG)* of Frederickson and McBryan. We expect the results of these studies to offer insight and tools both for the selection of new algorithms as well as for conversion of existing large codes for massively parallel architectures. We shall implement the programs on LLNL's Meiko-CS2 which is a distributed memory multiprocessor with 256 nodes, each node having two vector processors and its own communications processor.

The goal of this work is to advance the understanding of the fundamental concepts for the solution of systems of differential equations in a massively parallel environment. This solution process is key to DOE research and development programs. This work would contribute to the knowledge of the use of massively parallel distributed memory multigrid algorithms and several areas of current interest in theoretical astrophysics.

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, 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 product, 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 any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

17. Work Proposal Description

We shall (1) develop a *data communication manager* for distributed memory computers; (2) develop a *concurrent system/concurrent subgrid multigrid method* and compare this technique with standard parallel multigrid algorithms such as sequential subgrids and the *Parallel Superconvergent Multigrid (PSMG)* of Frederickson and McBryan. These algorithms will be designed for the Meiko CS2, a distributed memory vector multiprocessor computer. Results will be analyzed and compared.

The successful cooperation of n processors to solve a computational problem requires efficient communication of results between processors and efficient parallelization of the computation involved. We propose to develop tools to address these two implementation problems when converting a large sequential code to a distributed memory multiprocessing environment. We shall implement these tools in the three dimensional relativistic hydrodynamics program discussed below. Simulations from this code would contribute to the knowledge of the use of massively parallel distributed memory multigrid algorithms and significantly impact several areas of current interest in theoretical physics and astrophysics.

One of the greater challenges in utilizing parallel distributed memory computers is the conversion of existing software. Though parallelism may be readily identifiable in the computational parts of the program, their size and complexity prevent efficient conversion to newer distributed processing systems. The major problem in utilizing distributed memory processors is data management. Although researchers have studied optimizing data distribution to architecture and tasks, the practical problem of implementing these models into existing codes is not addressed. We propose to develop a *data communication manager* for distributed memory computers and to implement it in a three dimensional relativistic hydrodynamics program as an aid in program conversion to a parallel environment to address the problem of efficient communication.

Secondly, for efficient computation, we shall develop a *concurrent system/concurrent subgrid multigrid method*. Currently, five systems are approximated sequentially using multigrid successive overrelaxation. Results from an iteration cycle for one multigrid system are used in the multigrid iterations of the next set, and so on. We shall develop a concurrent multigrid algorithm for simultaneous computation of the sets of equations. In addition, we shall implement a method for concurrent processing of the subgrids in each of the multigrid computations. The conditions for convergence of the method will be examined. This method will be compared to standard parallel multigrid algorithms such as distributed-data-but-sequential subgrids, and the *Parallel Superconvergent Multigrid (PSMG)* of Frederickson and McBryan. We shall implement the programs on LLNL's Meiko-CS2 which is a general purpose distributed memory multiprocessor with 256 nodes, each node having two vector processors and its own communications processor. We expect the results of these studies to offer insight and tools both for the selection of new algorithms as well as for conversion of existing large codes for these types of architectures.

The work supported by this grant will involve conversion of a program which does state of the art modeling of multi-dimensional hydrodynamics, general relativity and particle transport in energetic astrophysical environments. One set of calculations currently

involves 8 hours of Cray YMP CPU time obtained over a period of about 10 days in a time shared environment. It represents the limit of what can be achieved in a single processor environment. Ultimately, dozens of such calculations must be done for dozens of parameter sets and initial conditions. Hence, a strong need exists for the increased through-put which a massively parallel environment can provide.

The main demand on CPU time arises from the many calls to an elliptic equation solver for the relativistic field equations. This accounts for about 80% of the CPU time. Clearly, the study and development of fast elliptic equation solvers for massively parallel machines could significantly benefit this program as well as the LLNL supercomputing initiative.

The goal of this work is to advance the understanding of the fundamental concepts for the solution of systems of differential equations in a massively parallel environment. This solution process is key to DOE research and development programs involving large scale hydrodynamic modeling. The strength of this proposal lies in the adaptation and integration of several areas of existing Lawrence Livermore National Laboratory expertise. One area is the application of parallel numeric methods. Another is the application of multi-dimensional hydrodynamics.

20.a Facilities

This work shall be done on LLNL's Meiko CS2 which is available through the Open Computing Facility (OCF) of LLNL. The Meiko CS2 is a general purpose MIMD supercomputer with both scalar and vector processing. It is a distributed memory system composed of "nodes". Each vector node has 1GB local disk, two 50Mhz Fujitsu microprocessors, 128MB memory, a SPARC super-scalar processor with 1MB second-level cache, and 70 Mhz Elan/Elite multistage interconnect. The processors cooperate by passing messages across a network. The Elan Communications Processor connects the processing elements to the network. The Elite switch is an 8-way crossbar switch allowing input/output without contention. (Meiko CS2 Technical Specifications, 1993, Meiko Corp.)

The Meiko CS2 at LLNL currently has 256 nodes on the open network. Eventually nodes will be moved to the Secure Computing Facility (SCF) and some will remain on OCF. The P.I. will have access to the SCF for testing purposes.

20.b Publications Pertaining to the Proposal

- Avila, J. H., and Gaffney, J., 1980, Technology Development Corporation Report.
Avila, J. H., and Tomlin, J. A., 1979, in *Proc. Computer Science & Statistics*, (University of Waterloo).
Bowers, R. L. & Wilson, J. R. 1985, *ApJ*, 295, 14
Corynen, G., Goodman, D., McDonald, J., Ng. L., Sholl, P. L., 1986, UCID-20924.
Handler, F., Sholl, P. L., Young, K. D., 1990, CD-90-0119.
Kaufman, A. M., Sholl, P. L., Johnson, K. C., 1990, CD-90-0169.
Mathews, G. J., Evans, C. R., and Wilson, J. R. 1986, in *Advances in Nuclear Astrophysics*, E. Vangioni *et al.* eds., (Editions Frontieres; France) p. 211

- Mathews, G. J., Wilson, J. R., Evans, C. R., and Detweiler, S. L. 1988, in *Proc. Texas A&M Symp. on Hot Nuclei*, S. Shlomo et al., eds, (World Scientific; Singapore), p. 24
- Mathews, G. J., Aufderheide, M. B., Wilson, J. R., Fuller, G. M., Jedamzik, K., Meyer, B. S., Woosley, S. E., and Hoffman, R. D., *New Production Mechanisms for Unstable Nuclei in Stars, Supernovae, and the Big Bang*, in *Radioactive Nuclear Beams - III*, D. J. Morrissey et al., ed. (1993) in press.
- Mayle, R. W. & Wilson, J. R. 1988, *ApJ*, 334, 909
- Mayle, R. W. & Wilson, J. R. 1991, in *Supernovae*, ed. S. E. Woosley, (New York: Springer), p. 333
- Mayle R. Schramm D. N., Turner, M. S., & Wilson, J. R. 1993 *Phys. Lett.*, B317, 119.
- Mayle, R. W., Travani, M., & Wilson, J. R. 1993 *Astrophys. J.*, 418, 398.
- Qian, Y.-Z., Fuller, G. M., Mathews, G. J., and Wilson, J. R. 1993, *Supernova-Nucleosynthesis Constraints on Cosmologically Significant Neutrino Masses and Mixing Angles*, *Phys. Rev. Lett.* 71, 1965 (1993).
- Sholl, P. L., 1990, LLNL Report UCRL-JC-10374.
- Sholl, P. L., 1989, LLNL Report CD-89-0247.
- Sholl, P. L., 1988, LLNL Report UCRL-99216.
- Sholl, P. L., 1988, LLNL Report UCID-21389.
- Sholl, P. L., 1988, LLNL Report UCID-21408.
- Sholl, P. L., and Wolfe, R. K., 1985, *Computers and Industrial Engineering*, V. 9, No. 3, 247.
- Wilson, J. R. 1972, *ApJ*, 173, 431
- Wilson, J. R. 1975, *Ann. N. Y. Acad. Sci.*, 262, 123
- Wilson, J. R. 1977, in *Proc. Marcel Grossman Meeting*, ed. R. Ruffini (Amsterdam; North Holland) p. 393
- Wilson, J. R. 1978, in *Proc. Int. School Phys., Fermi Course LXV*, ed. R. Giacconi & R. Ruffini (Amsterdam; North Holland) p. 644
- Wilson, J. R. & Mathews, G. J. 1989, in *Frontiers in Numerical Relativity*, C. R. Evans, L. S. Finn, & D. W. Hobill, eds., (Cambridge University Press; Cambridge) p. 306
- Wilson, J. R. & Mathews, G. J. 1994, in *Proc. 7th Marcel Grossman Meeting on General Relativity* (World Scientific; Singapore) in press.
- Wilson, J. R. & Mayle, R. W. 1988, *Phys. Rep.*, 163, 63.
- Wilson, J. R. & Mayle, R. W. 1993, *Phys. Rep.*, 227, 97.
- Woosley, S. E., Wilson, J. R., Mathews, G. J., Hoffman, R. D., and Meyer, B. S., *The r-process and Neutrino-Heated Supernova Ejecta*, *Astrophysical J.*, in press (1994), UCRL-JC-115299.

20.c. Purpose

20.c.1. Data management software for porting large codes to distributed-memory computers

One of the greater challenges in utilizing parallel distributed-memory computers is the conversion of existing software. Many programs have been growing for years. They are of sufficient size and complexity that complete revision would require unacceptable costs to "port" them to newer distributed systems. This is true despite the fact that parallelism is readily identifiable in the computational parts of the program. The major problem in utilizing distributed memory processors is data management; a considerable effort must be made to map the required data to the individual processors. The difficulty is that each processor modifies a portion of the problem data and other processors will subsequently need to get the new data from the appropriate processor.

Although much work has been done on the efficient distribution of data to architecture and tasks, these algorithms do not address the practical problems encountered when converting an existing large code with complicated interrelationships of data without totally rewriting the code. If one is modifying a large program, it is very difficult to know the interrelationships of various working arrays used to carry forward the calculations, how their modification affects other parts of the calculations and which items have been modified and need to be updated. Moreover, the location of modified items needs to be tracked so that updated copies can be requested from the appropriate processors. Preliminary approaches to data management in a distributed environment often leave the task to the operating system or to compilers. We feel that approach, for now, is hopelessly inefficient for these applications. These large programs, prime candidates for parallel processing, may have a million lines of code put together over years by several programmers. In addition, the conversion itself is usually done by several programmers who need to worry about how their changes affect other parts. A possible solution would be to conservatively assume that all data that is required for the computation of problem variables has been modified and to request "clean" copies. In the program we propose to convert, this would require sending over 30MB of data to 64 or more processors. Unfortunately, the communication time required to transfer large amounts of data between processors is generally large. Also when most scientific applications involve iteration to obtain a solution, this conservative solution has a prohibitively high cost in running time.

We propose to develop an alternate strategy to assist in the solution of this problem. This strategy involves developing data-management (DM) software to determine what to send, where to find it, who to send it to, and actuating transmittal. Thus, communication (data-passing) time is minimized by sending only the data needed for a computation, only to those processors requiring it, and only if that data has changed since the last communication with the processor. These tasks will be done by the DM software, relieving the programmer of this responsibility. This will greatly simplify the programming of distributed memory machines and make the programming effort similar to that of a shared-memory system.

This DM software will be implemented in the following way. At the start of a parallel section of code, each processor requests the latest copies of those variables that it requires to do its computations. The DM software accepts these requests and coordinates the appropriate message traffic to satisfy all processors. This DM software will keep track of the location of data in the distributed system and will coordinate the transfer of that data to those processors that require it. Moreover, the DM software will also keep track

of the time when data is modified so that requests for data that have not been modified since they were last acquired do not result in unnecessary data transmission. This averts transmitting any data items that have not changed since the last time a processor acquired them. At the end of a parallel section of code, each processor transmits a list of those items that have been modified (but not the items themselves). The DM software records the location of these modified items as well as the time at which they were modified.

The benefit to the programmer is that he or she can concentrate more on the parallelism and less on the data transfer portions of the code. Moreover, the programmer can concentrate on splitting up the computations without having to be fully aware of the intricacies of other portions of the program data dependencies impacting the local computations.

This general purpose memory manager for distributed systems will be a useful tool to simplify the programming or reprogramming task by reducing the problems associated with data interactions. This would aid in bringing the cost of programming down and the use of such systems by more organizations.

We propose to develop this software and to explore its impact on a multigrid code that models the physics of a coalescing binary neutron star system. This program currently requires hours of time on a Cray YMP system to obtain a solution. We shall develop the DM software on a Meiko CS-2 system and report on the results of porting the 3-D relativistic binary neutron star code to this system. The program will be general enough to be widely useful in other large multi-dimensional hydrodynamics and transport physics codes and other distributed memory computers.

20.c.2. Distributed Memory Parallel Algorithms for Solving a System of Elliptic Partial Differential Equations

The second part of the proposal is the development and testing of distributed memory algorithms for the multigrid successive overrelaxation numerical method currently used in the program. The standard MG relaxation methods are heavily sequential, that is, the unknowns arise from a sequence of nested grids. This creates some difficulties when trying to parallelize the algorithms because the iteration on one grid depends on results from another grid. Adaptations of this basic model have been made for parallel implementation on machines with moderate numbers of processors. However, today the fastest supercomputers are massively parallel. One might expect that the application of multigrid method on these systems would be straightforward, extrapolating from the existing work on parallel computers. Unfortunately this has not been the case. Machines with massive parallelism have not been able to achieve the high efficiency rates attained by machines with moderate parallelism for these methods mainly because of the inability to fully utilize the processors in the computation on coarse grids. See, for example, results and problems of implementing the standard multigrid algorithm on the CM-1 discussed in McBryan (1987b).

Though there has been a significant number of studies in parallel multigrid methods, the majority of this work is concerned with the performance of parallel implementations of sequential grids on moderate sized multiprocessors or on hypercube computers. We propose to look at the implementation issues for a more general architecture distributed processing system. In addition, we shall investigate the issue of concurrent iterations of the

subgrids in order to achieve significant speed-up and efficiency on this massively parallel system. We shall develop the software on a Meiko CS-2 system. One of the advantages of the Meiko architecture is the opportunity to combine vector processing at an inner level with distributed computing at an outer level. Being able to utilize both of these forms of parallelism simultaneously offers unique possibilities. We shall compare execution times with the Cray YMP which currently runs the code.

Parallel processing methodology is still evolving. The full potential of parallel computation will only be realized when the software, numerical algorithms, and computational strategies combine to effectively use the hardware.

20.c.3. Binary Neutron Star Coalescence Simulations

In this program we are developing the first necessary tool to apply to the problem of this possible source for gravity waves and cosmological gamma-ray bursts, which have lacked a definitive calculation for their origin and evolution. Ultimately this code will be used to follow the coalescence of a neutron-star binary system. It will give the first fully relativistic 3-dimensional information of the ensuing gravity wave signal and possible gamma-ray burst specific internal energy density. These results could significantly impact current astrophysical theory.

A more detailed discussion of various aspects of the proposed work follows in the background discussions of Sec. 20.d.

20.d Background

20.d.1 Massively Parallel Processors

The desire to solve more and more complex problems outstrips the capabilities of available computing time on existing hardware. The need for fast computations often arise in the solution of partial differential equations. Examples include computational fluid dynamics, weather prediction, image processing, and nuclear weapons design and testing. Many computer programs at DOE's National Laboratories can take days of Cray time to execute, causing limitations or delays in some research programs due to computational costs or the availability of computer time. Hardware advances have raised the possibility of massively parallel computation to enable the solution of such problems, but has also created new computational needs in the development of parallel algorithms to utilize the hardware potential.

Computer programs and numerical algorithms have to be modified or restructured to utilize special computer architectures. Shared memory parallel computers with a moderate number of processors have a programming model that is much more closely aligned with the traditional single processor approach and so are easier to program. On the other hand, distributed memory architectures have the advantage of being able to use hundreds or thousands of processors. But the trade-off is in managing the communication between processors. The mapping of the algorithm onto such architectures can often involve an extensive reprogramming overhead. Special tools are necessary to coordinate and manage

communication between processors in a distributed processor environment. Moreover, programming shared memory multiprocessors has been described as an "evolutionary" approach to parallel programming in contrast to a "revolutionary" approach for distributed memory processors.

Although much work has been done on efficient distribution of data to tasks (Upfal & Wigderson, 1984; Wholey, 1990; Nitzberg and Lo, 1991; Chrisochoides et al, 1994; Sinharoy and Szymanski, 1994; and others), these algorithms do not address the practical problems encountered when converting an existing large code with complicated interrelationships of data without totally rewriting the code. If one is modifying a large program, it is very difficult to know the interrelationships of various working arrays used to carry forward the calculations. One must be aware of how their modification affects other parts of the calculations and which items have been modified and need to be updated. Moreover, the location of modified items needs to be tracked so that updated copies can be requested from the appropriate processors. We shall develop data-management (DM) software to do the job of what to send, where to find it, who to send it to, and actuating transmittal. This general purpose memory manager for distributed systems would be a useful tool to simplify the programming or reprogramming task by reducing the problems associated with data interactions.

20.d.2 Parallel Multigrid Background

A detailed discussion of the classical multigrid method from the point of view of a parallel realization can be found in Brandt (1981). Though MG methods have a high degree of parallelism, one drawback is that with most parallel implementations, large numbers of processors cannot be fully utilized. This is mainly due to the fact that the full use of the processor array attained during the computation in the finest grid is not maintained over coarser grids. However, there has been a significant number of studies in parallel multigrid methods with a great deal of success.

The types of studies on parallel implementations that have emerged fall into two categories:

1. Sequential grids with computations distributed among processors. Adaptations of this include attempts to accelerate convergence by decreasing the number of iterations required for convergence (Douglas & Miranker, 1987; Hackbusch, 1987; Tuminaro & Chan, 1987; Chen & Sameh, 1989; Tuminaro, 1989; and others). Making the proper choices of the operators (interpolation and restriction, for example) is critical to effective decomposition of the problem.
2. Concurrent iteration of the grids (Gannon & van Rosendale, 1982; Greenbaum, 1985; Frederickson & McBryan, 1988). Here, all levels are processed simultaneously. Essentially, the original problem is distributed over all grids, relaxation sweeps are done on all levels, and then the solutions are recombined. Since grid computations are simultaneous, one iteration should be much faster than a single iteration of a standard multigrid method.

The majority of the work that has been done is concerned with the performance of parallel implementations of sequential grids on computers with a moderate number of

processors or on hypercube computers (for example, Chan & Saad, 1986; McBryan & Van de Velde, 1986; Briggs et al 1987; Chan & Tuminaro, 1987a; Chan & Tuminaro, 1987b; McBryan, 1987a; Thole, 1988). Though one might expect it would be straightforward to extrapolate to massively parallel computers, this has not been the case. Machines with massive parallelism have not been able to achieve the high efficiency rates attained by machines with moderate parallelism for these methods.

We shall investigate data mapping, interpolation and restriction operators of the multi-grid algorithm on new architectures for our implementation on the Meiko CS2. But also, to increase speed-up and efficient use of the larger number of available processors, we examine the use of concurrent subgrid methods and concurrent systems approximation on this type of parallel architecture. In regards to this, we review three of the most promising models of the concurrent subgrid computation type: Gannon & van Rosendale (1982); Greenbaum (1985); and PSMG (Frederickson & McBryan, 1988). This is followed by a description of our proposed work.

Gannon & van Rosendale illustrated a complete parallelization of a cycle by a suitable decomposition of the right-hand-side (RHS) of the approximated elliptic system and by parallel organization of data transfers between grids. However there are significant problems with their algorithm. Implementation of the algorithm is complicated. Convergence rates are somewhat slower than the corresponding multigrid method and are not bounded independent of the number of mesh points. Its efficiency is only suggested by numerical experiments. For hypercube computers, there is no efficient mapping of grids to processors (Chan & Saad, 1986).

In the Greenbaum model, the residual vector is computed over the finest grid in every cycle and is projected onto coarser grids. This is then used as the RHS for the systems of linear equations. These equations are then approximately solved concurrently using relaxation on each grid. The results are combined to obtain a new approximate solution. Greenbaum implemented and tested the model on a CRAY XMP48 using 4 processors and 4 grids.

The amount of work per processor depends on how the work is assigned to individual processors. Maximal parallelism is achieved by an organization of the processing in which all the computation phases are uniformly distributed to all processors. In this case, synchronization of the whole process presents a serious problem. Thus the application of a single processor for all operations over a given grid may turn out to be more effective. Convergence is slower. Experiments performed on the CRAY XMP showed that, in view of the high synchronization costs and computational power, the CRAY XMP48 is not suited for solving problems over coarse grids (Greenbaum, 1985).

It also appears that the above concurrent iteration approach has no efficient mapping of grids onto separate processors of a hypercube (Tuminaro, 1989). However, we shall explore mappings of these methods to the Meiko architecture.

An approach which may overcome the difficulty of keeping a large portion of processors active on coarse grids also is developed in Fredrickson & McBryan (1988). The Frederickson and McBryan PSMG (Parallel Superconvergent Multigrid) method allows one to solve several coarse grid problems concurrently. Frederickson & McBryan's idea is to do multiple coarse grid corrections instead of one and combine these multiple coarse grid corrections

to improve the convergence rate of the MG algorithm. The different coarse grid equations solve the residual equation in different subspaces arranged such that aliasing errors cancel when the subproblem solutions are combined, thus accelerating convergence.

Vajtersic (1993) states that the PSMG may be the only truly "massively" parallel MG algorithm to-date. However, Tuminaro & Chan (1989) showed that the choice of the restriction and prolongation operators is critical to the convergence rate. If the wrong operators are used, convergence is slower and one may even find that the method will diverge.

Other studies (Chan & Tuminaro, 1990; Decker, 1990) indicate that, although the convergence rate is good, the relaxation scheme employed in the PSMG is relatively expensive in both communication and computation costs. Computation and communication costs are dependent on the machine architecture, the algorithm, and the PDE. Tuminaro and Chan (1989) have proposed modifications of the original PSMG which use less expensive relaxations. These modifications speed up convergence but they conclude that more research is needed to determine principles which indicate how inexpensive operators should be chosen to achieve rapid convergence. On the other hand, Decker (1990) indicates that the modifications of the original PSMG using the less expensive relaxations proposed by Chan & Tuminaro gives only limited improvement over the standard multigrid methods and that most of the processors are doing useless work on the coarse grids. Decker further states that the PSMG algorithm, though it achieves perfect processor utilization, is no more efficient than a parallel implementation of standard MG methods, "PSMG is not significantly better than a reasonably efficient parallelized version of the standard MG algorithm and is just as limited by the inherent constraints of the MG technique as the standard algorithms are". (See also Decker, 1991; Frederickson & McBryan, 1991.)

It is clear that parallel processing methodology is still evolving. The full potential of parallel computation will only be realized when the software, numerical algorithms, and computational strategies combine to effectively use the hardware.

In the program we propose to convert, the elliptic equations for the field, and the differential evolution equations for the hydrodynamics variables, were finite-differenced in a Cartesian x,y,z isotropic coordinate system with rotating, nonuniform grids. After finite-differencing, the elliptic equations can be reduced to matrix equations of the form $Mx=b$. M is sparse, and x is a vector representing the relevant field variable at each zone and b is derived from the source term. Though there are intermediate calculations, there are five sets of matrix equations that are solved using the multigrid successive overrelaxation method. These are for the conformal factor (eqn. 4 of next section), the conformal factor multiplied by the lapse function (eqn. 6 of next section), and the three shift vectors in the x,y , and z directions (eqn. 7 of next section). Note, each of the shift factors has its own set of elliptic equations to be solved; the shift factors are the shift in space-like coordinates between hypersurfaces. These five variables, on the finest grid, are currently of size $100 \times 50 \times 50$ each. Currently the sets of equations for the lapse function and the conformal factor are solved, the results then used in the calculations of the three sets of shift vector elliptic equations.

The approach we plan to investigate is to distribute each of a set of five elliptic systems over the processors and solve them concurrently. There are two approaches we

plan to pursue to handle the data dependencies in the concurrent calculations. One is sequential subgrid calculations within each multigrid set of calculations. The other is concurrent subgrid computations within each set of multigrid calculations being performed concurrently.

In the first (sequential subgrids), those sets of equations which use values from the other sets would be passed to the grids of a variable being approximated along with those interpolated or restricted values of that variables calculation. Synchronization is important.

The other approach, concurrent subgrid calculations on each set of elliptic equations would progress similar to that outlined by Greenbaum. Errors from the finest grid would be projected onto each of the subgrids. At the same time, these projections for the different subgrids would be passed to the appropriate level of subgrid calculation on the given set of elliptic equations. It is expected, though, that some of the problems encountered when Greenbaum implemented her method on the Cray XMP can be overcome on the Meiko. The use of processors doing coarse grid calculations can be made more efficient by having those processors working on coarse grid calculations for several sets of the elliptic equations. Again, though, synchronization is important. This shall be handled by our data manager.

Finally, these two approaches will be compared to results using the Frederickson-McBryan PSMG, which has attained good convergence and efficiency in several applications on hypercube and Connection machines.

20.d.3 Relativistic Binary Neutron Star Coalescence

Here we discuss the (3+1) dimensional general relativistic hydrodynamics code Wilson & Mathews, 1994) to which we will apply the algorithms developed in this proposal. In the code, the relativistic field equations are solved exactly at each time slice with a constraint that the spatial 3-metric remains conformally flat. The effect of gravitational radiation is introduced by a multipole radiation reaction formula.

For much of the evolution of a neutron star binary system, the two neutron stars are expansions of the point sources using post-Newtonian techniques (Lincoln & Will 1990; Cutler et al. 1993). However, as the stars approach one another the effects of the strong gravitational field and the hydrodynamics of the merger are expected to dominate the evolution of the system causing the wave forms to become quite complex. This complexity, however, will carry information about the coalescing system (Cutler et al. 1993) so that careful modeling is needed for extraction of the information contained in the gravity waves. A computation of the hydrodynamic and gravitational evolution is complicated, though, due to the inherently three-dimensional character of the orbiting system. In the present work we therefore have attempted to explicitly include most of the effects of a fully general relativistic hydrodynamic treatment. In essence, we solve the exact general relativistic equations at each time slice and advance the fluid variables in the previously found metric. The gravitational radiation reaction is then added in a multipole expansion. The justification for this is that the radiation reaction is small, and for most of the evolution, the wavelength of the radiation is much larger than the size of the system.

We have selected Cartesian x, y, z isotropic coordinates with the usual ADM (Arnowitt, et al, 1962) slicing of spacetime into a one-parameter family of hypersurfaces separated by

differential displacements in time-like coordinates. Thus, proper distance is expressed

$$ds^2 = -(\alpha^2 - \beta_i \beta^i) dt^2 + 2\beta_i dx^i dt + \gamma_{ij} dx^i dx^j, \quad (1)$$

where α is the lapse function and β_i is the shift in space-like coordinates between hypersurfaces.

We introduce *the conformally flat approximation* or *CFA*, motivated by the general observation that gravitational radiation in most systems is small

$$\gamma_{jk} = \phi^4 \delta_{jk}, \quad (2)$$

where ϕ is the conformal factor describing the overall curvature of the 3-geometry and δ_{jk} is the Kronecker delta.

From the exact equation (York 1979) for the evolution of the 3-metric, a conformally flat 3-metric requires,

$$2\alpha K_{ab} = (D_a \beta_b + D_b \beta_a - \frac{2}{3} \gamma_{ab} D_c \beta^c), \quad (3)$$

where D_a is the covariant derivative in the three space, and K_{ab} is the extrinsic curvature. As a gauge choice we also select maximal slicing, $\text{tr}(K_{ab}) = 0$.

The Hamiltonian constraint equation (York 1979) reduces to a flat-spaced Laplacian for the conformal factor,

$$\nabla^2 \phi = -4\pi \rho_1 \phi, \quad (4)$$

where ρ_1 can be written in terms of the physical hydrodynamic variables,

$$\rho_1 = \frac{\phi^4}{2} \left[DW + \left(\Gamma W - \frac{(\Gamma - 1)}{W} \right) + \frac{1}{16\pi} K_{ij} K^{ij} \right]. \quad (5)$$

We also use the Hamiltonian constraint and the maximal slicing condition, $\text{tr}(K^{ij}) = 0$, to obtain an elliptic constraint equation for the lapse function α .

$$\nabla^2(\alpha\phi) = 4\pi(\alpha\phi) \frac{\phi^4}{2} \left[\frac{D(3W^2 - 2) + E[3\Gamma(W^2 + 1) - 5]}{W} + \frac{7}{16\pi} K_{ij} K^{ij} \right]. \quad (6)$$

The momentum constraint reduces to an elliptic equation for the shift vector.

$$\nabla^2 \beta^i = 16\pi \left[\alpha S_i - \beta^j W \left(D + \Gamma E \right) \right] + \frac{\phi^6}{\alpha} \frac{\partial(\alpha/\phi^6)}{\partial x^j} \left(\frac{\partial \beta^i}{\partial x^j} + \frac{\partial \beta^j}{\partial x^i} - \frac{2}{3} \delta_{ij} \frac{\partial \beta^k}{\partial x^k} \right) - \frac{1}{3} \frac{\partial}{\partial x^i} \frac{\partial \beta^k}{\partial x^k}. \quad (7)$$

Most of the momentum encompassed in equation (7) is simply the orbital motion of the binary. Therefore, we decompose $\beta^i = G^i + (\omega \times r)^i$ into a frame-drag term, G^i and an orbital motion term in which the non-orbital aspects of matter evolution and relativity can be studied more easily. We subtract the orbital velocity of the coordinate system from both sides of equation (7) since $\nabla^2(\omega \times r) = 0$, β^i and G^i can be used interchangeably.

The Eulerian relativistic hydrodynamic equations can be written (York 1979) in a form which is reminiscent of their Newtonian counterparts in terms of the coordinate covariant density, $D = W\rho$, the energy, $E = W\rho\epsilon$, the three velocity, $V^i = \frac{U_i}{\alpha W} - \beta^i$, and the momentum density, $S_i = (D + \Gamma E)U_i$, where U_i is the four-velocity. An additional relativistic contribution to the hydrodynamic evolution comes from the effects of the radiation reaction potential on the momentum density which at the quadrupole level is just

$$\left(\frac{\partial S_i}{\partial t}\right)_{RR} = -\alpha W(D + \Gamma E) \frac{\partial}{\partial x^i} \left(\frac{\partial^5 Q_{ij}}{\partial t} x^i x^j \right), \quad (8)$$

The quadrupole moment Q_{ij} , and other moments, is well defined in our CFA and can be extracted (Thorne, 1979) from the boundary values of source term for the conformal factor, ρ_1 .

$$Q_{ij} = 2 \int dV \rho_1 \left[x_i x_j - \frac{1}{3} \delta_{ij} \right]. \quad (9)$$

We presently restrict our consideration to the dominant radiation due to orbital motion. The loss rate of angular momentum from the system due to gravity waves is then,

$$\frac{dJ}{dt} = \frac{2}{5} \omega^5 Q^2, \quad (10)$$

where, ω is the rotational frequency of our coordinate system. The measured transverse travelless gravity wave metric perturbation is

$$h = 2 \frac{\omega^2 Q}{r} \sin 2\omega t, \quad (11)$$

As an illustrative first calculation we have placed the stars at various separation distances on the grid and only run long enough to check that the system is in pseudo equilibrium. We initially solved the TOV equations for individual $\sim M_\odot$ $\Gamma = 2$ neutron stars and mapped them directly to our coordinates. The hydrodynamic and field equations were then evolved until the stars found a new quasiequilibrium configuration in our coordinates. Figure 1a shows an example of density contours of an orbiting binary after about 700 cycles (a significant fraction of an orbit). Note that the stars relaxed to nearly spherical contours at this separation distance.

From this quasiequilibrium, we then perturb the angular momentum and allow the stars to find a new stable orbit. Figure 1b shows the evolution of the distance between centers of two neutron stars for about 8 ms. The angular momentum was perturbed after about 4 ms. After the orbit has stabilized we extracted the quadrupole moment, orbital frequency, binding energy and angular momentum. From the time integration of this we can estimate the gravity wave signal to be expected. In the future we plan to complete this picture with many more runs and to follow details of the last few orbits of coalescence. We can also eventually make studies of mass and equation-of-state dependencies of the gravity wave signal.

The calculations shown in figures 1a,b are the result of about 8 hours of Cray YMP CPU time obtained over a period of about 10 days in a time shared environment. It represents the limit of what can be achieved in a single processor environment. Ultimately,

dozens of such calculations must be done for dozens of parameter sets and initial conditions. Hence, a strong need exists for the increased through-put which a massively parallel environment can provide.

The main demand on CPU time for the calculation shown in figs.1a,b comes from the many calls to the elliptic equation solver for the relativistic field equations. This accounts for about 80% of the CPU time. Clearly, the proposed parallel algorithm development in this proposal, particularly the study and development of fast elliptic equation solvers, could significantly benefit this program.

References

- Arnowitt, R., Deser, S. and Misner, C. W., 1962, in *Gravitation*, ed. L. Witten (New York: Wiley), p. 227.
- Bailey, D.H., 1993, Experience with Parallel Computers at NASA Ames, *Intl. J. of High Speed Computing*, V. 5,N. 1 (1993), pp 51-62.
- Bradaschia, C., *et al.*, 1990, *Nucl. Instrum. Meth., Phys. Res. Sect.*, 289, 518.
- Brandt, A., 1981, Multigrid solvers on parallel computers, in *Elliptic Problem Solvers*, M. H. Schultz Ed. NY: Academic, 1981, pp. 39-83.
- Briggs, B., Hart, L., McCormick, S. and Quinlan, D., 1987, Multigrid methods on a hypercube, in *Proc. of the Third Copper Mountain Conference on Multigrid Methods*, S. McCormick (ed), M. Dekker, NY, 1987, pp63-84.
- Chan, T. F. and Saad, Youcef, 1986, Multigrid Algorithms on the Hypercube Multiprocessor, *IEEE Transactions on Computers*, V. C-35, N. 11, Nov. 1986.
- Chan, T.F. and Tuminaro, R.S., 1987a, A Survey of Parallel Multigrid Algorithms, in *Parallel Computations and Their Impact on Mechanics*, A.K. Noor, Ed. NY: ASME, 1987, pp. 155-170.
- 1987b, "Multigrid algorithms on hypercube processors", in *Proc. of the Second Conf. on Hypercube Multiprocessors*, M. Heath,ed., SIAM, Philadelphia, 1987.

pp730-737.

— 1989, Analysis of a parallel multigrid algorithm, in *Proc. of the Fourth Copper Mountain Conference on Multigrid Methods*, J. Mandel and S. McCormick eds.

Chen, H. and Sameh, A., 1989, "A matrix decomposition method for orthotropic elasticity problems", *SIAM J. Matrix Anal. Appl.*, 10 (1989), pp.17-17.

Chrisochoides, N., Houstis, El and Rice, J., 1994, Mapping Algorithms and Software Environment for Data Parallel PDE Iterative Solvers, *J. Parallel & Distributed Computing* 21:75-95.

Cutler, C., *et al.*, 1993, *Phys Rev. Lett.*, **70**, 2984.

Decker, N.H., 1990, On the parallel efficiency of the Frederickson-McBryan Multigrid Algorithm, NASA CR-181999.

— 1991, *SIAM J. Sci. Stat. Comput.*, V12, N. 1, pp.208-220.

Douglas, C. and Miranker, W., 1987, "Constructive interference in parallel algorithms", *SIAM J. on Numerical Analysis*, 25 (1987), pp. 376-398.

Frederickson, P.O. and McBryan, O., 1988, Parallel superconvergen Multigrid, in S. McCormick (ed), *Multigrid Methods: Theory, Applications and Supercomputing*, Marcel Dekker, NY, 1988.

— 1991, *SIAM J. Sci. Stat. Comput.*, V. 12, N. 1, pp. 221-229.

Gannon, D. and Van Rosendale, 1986, J., On the Structure of Parallelism in a Highly Concurrent PDE Solver, *J. of Parallel & Distributed Computing* 3, 106-135.

Greenbaum, A., 1985, A Multigrid Method for Multiprocessors, Preprint UCRL-92211, LLNL, Livermore, 1985.

Hackbusch, W., 1987, "A new approach to robust multigrid methods", in First International Conf. on Industrial and Applied Mathematics, Paris.

Jalby, W., Frailong, J.M. and Lenfant, J., 1984, Diamond schemes: An organization of parallel memories for efficient array processing, Rapport de Recherche INRIA 342, INRIA, Rocquencourt.

Johnson, G. M., Swisshelm, J.M. and Kumar, S.P., 1985, Concurrent-processing adaptations of a multiple-grid algorithm, in Proc AIAA 7th Computational Fluid Dynamics Conference, Am. Inst. of Aeronautics and Astronautics, Inc., Cambridge, Mass., 1985, pp. 171-181.

Kolp, O. and Mierendorff, H., 1985, Efficient multigrid algorithms for locally constrained parallel systems, in Proc. 2nd Copper Mountain Conf. on Multigrid Methods, Colorado.

Lincoln, C. W. and Will, C. M., 1990, *Phys. Rev.*, **D42**, 1123.

Lonsdale, G. and Schuller, A., 1993, Multigrid efficiency for complex flow simulations on distributed memory machines, *Parallel Computing* 19 (1993) pp.23-32. North-Holland.

Marinescu, D.C., Rice, J.R. and Vavalis, E.A., 1993, Performance of iterative methods for distributed memory machines, *Applied Numerical Mathematics* 12:421-430.

McBryan, O., 1987a "The Connection Machine: PDE Solution on 65536 Processors", Tech. Rep. LA-UR-86-4218, Los Alamos National Laboratory, 1987.

— 1987b, "Numerical computation on massively parallel hypercubes", in M.T. Heath(ed), *Hypercube Multiprocessors* 1987, SIAM, Philadelphia, 1987, pp. 706-719.

- Wilson, J. R., 1979, in *Sources of Gravitational Radiation*, ed. L. Smarr (Cambridge; Cambridge Univ. Press) p. 423.
- Wilson, J. R. 1972, ApJ, 173, 431.
- Wilson, J. R. 1975, Ann. N. Y. Acad. Sci., 262, 123.
- Wilson, J. R. and Mayle, R. W., 1993, *Physics Reports*, 227, 97-111.
- Wilson, J.R., Mayle, R.W., 1993, to appear in *Physics Reports*.
- Wolszczan, A., 1991, Nature, 350, 688.
- York, J. W. Jr., 1979, in *Sources of Gravitational Radiation*, ed. L. Smarr (Cambridge; Cambridge Univ. Press) p. 83.

20.e Approach: Management Controls and Methodology

The principle responsibility for the overall management of the various aspects of this proposal is with the P.I., Patricia Sholl. We are requesting 2/3 FTE support for her contribution to this effort. In addition support for three collaborators is requested. We are requesting consulting support for John Avila (San Jose State University) who will be advising on the data manager as well as numerical methods. We also request support and some travel for G. Mathews (University of Notre Dame) as well as 0.25 FTE support for J. R. Wilson who will advise and conduct the physics applications.

The completion of significant advances toward the goals described in the background discussions of Sec. 20.d would mark the success of this proposal.

- McBryan, O.A. and Van de Velde, E.F., 1986, Elliptic Equation Algorithms on Parallel Computers, Communications in Applied Numerical Methods, V.2 pp.311-316.
- 1986, "Multigrid methods on parallel processors", in Multigrid Methods II, W. Hackbusch & U. Trottenberg (eds), Springer-Verlag, Berlin, 1986, pp.232-260.
- 1987, Hypercube Algorithms and Implementations, SIAM J. Sci. Stat. Comput. V. 8, N. 2, March 1987, pp. s227-s287.
- McCormick, S.F., 1989, *Multilevel Adaptive Methods for Partial Differential Equations*, SIAM, Philadelphia.
- Misner, C. W., Thorne, K. S. and Wheeler, J. A., 1973, *Gravitation*, (San Francisco: W. H. Freeman and Co.).
- Nitzberg, B. and Lo, V., 1991, Distributed Shared Memory: A Survey of Issues and Algorithms, IEEE Trans. on Comp., Aug. 1991, pp. 52-60.
- Piran, T. 1992, ApJ Lett., 389, L45
- Press, W. H. and Thorne, K. S., 1972, Ann. Rev. Astron. Astrophys., 10, 335.
- Qian, Y. Z., Fuller, G. M., Mathews, G. J., and Wilson, J. R. 1993, Phys. Rev. Lett. 71, 1965 (1993).
- Rasio, F. A. and Shapiro, S. L., 1992, Astrophys. J., 401, 226.
- Ritzdorf, H. and Stuben, K., 1994, Adaptive Multigrid on Distributed Memory Computers, in *Multigrid Methods IV*, P.W. Hemker and P. Wesseling (eds), Birkhauser Verlag, Berlin, pp. 77-109.
- Schutz, B. F., 1986, Nature, 323, 310.
- Schutz, B. F., 1989, Classical Quantum Gravity, 6, 1761.
- Sinharoy, B. and Szymanski, B.K., 1994, Data and Task Alignment in Distributed Memory Architectures, J. of Parallel & Distributed Computing 21:61-74.
- Thole, C. and Trottenberg, U., 1988, A Short Note on Standard Parallel Multigrid Algorithms for 3D Problems, Applied Math & Computation 27:101-115.
- Thorne, K., 1979, in Rev. Mod. Phys., 53, 299.
- Tuminaro, R.S., 1989, Multigrid Algorithms on Parallel Processing Systems, Stanford PhD. Dissertation.
- 1992, A Highly Parallel Multigrid-Like Method for the Solution of the Euler Equations, SIAM, J. Sci Stat. Comput., V.13, N. 1, pp.88-108.
- Tuminaro, R.S. and Womble, D.E., 1992, Analysis of the Multigrid FMV Cycle on Large-Scale Parallel Machines, SAND91-2199J, SNL, Albuquerque, NM.
- Upfal, E. and Wigderson, Avi, 1984, How to Share Memory in a Distributed System, Stanford Report No. STAN-CS-84-1024.
- Vajtersic, M., 1993, *Algorithms for Elliptic Problems*, Kluwer Academic Publishers, Czech Republic.
- Wholey, S., 1990, Automatic Data Mapping for Distributed-Memory Parallel Computers, ACM, 1990, pp. 25-34.
- Wilson, J. R. and Mathews, G. J. 1994, in *Proc. of the Seventh Marcel Grossman Meeting on General Relativity*, in press.
- Wilson, J. R., 1979, in *Sources of Gravitational Radiation*, ed. L. Smarr (Cambridge; Cambridge Univ. Press) p. 423.
- Wilson, J. R., 1977, in *Proc. Marcel Grossman Meeting*, ed. R. Rufini.