CANADIAN JOURNAL OF EXPLORATION GEOPHYSICS
VOL. 32, NO. 2 (DECEMBER 1996), P. 121-129

HIGH PERFORMANCE COMPUTATION (TERAFLOPS) FOR SEISMIC PROCESSING

R. PHILLIP BORDING[1]

ABSTRACT

Many fundamental seismic processes are based on wave equations. The finite difference formulation of the acoustic wave equation is transformed into a compute engine, a wave equation difference engine. This massively parallel compute engine is truly scalable and does not exhibit the poor performance characteristics of existing parallel machines. The prototype seismic modeling machine is the basis for an imaging wave machine which uses a reverse time algorithm. This pre-stack algorithm in hardware is capable of supporting teraflop execution rates. This makes it feasible to compute 3D pre-stack depth migrations in one day.

WAVE MACHINES INTRODUCTION

Machines are mans' way of reducing work, saving human effort, and in general making life more enjoyable. The use of digital computers as a tool for problem solving has expanded to all aspects of modern life. Prior to the digital era many computational machines were built to aid in problem solving. The sextant and slide rule are good examples. Machines were built to compute latitude, tell time, and to predict the tides. These mechanical devices reached a remarkable state of refinement over the centuries of the industrial revolution. The development of interchangeable parts, a by-product of mass production helped Charles Babbage (Moseley, 1964) in his effort to build the first digital computer, the Difference Engine, shown in Figure 1. This pioneering engineering project of the early 1800s attempted to construct a mathematical difference engine, to compute Newton's difference calculus. It did function as designed but was plagued by friction and mechanical troubles.

Just recently a new machine was built to celebrate his birthday using his designs and it worked. This replica of the Babbage Difference Engine was built using the same machine tools of his era and benefited from a careful study of his designs. Surely he would have been pleased to see it operate.

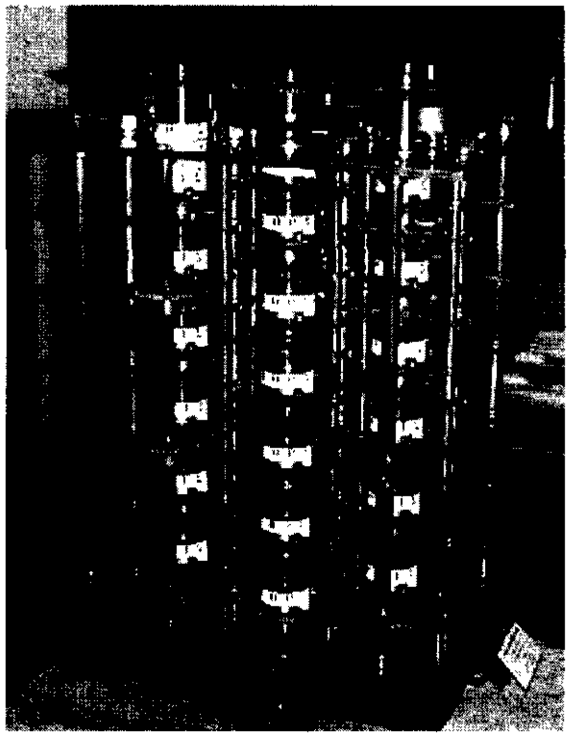


Fig. 1. The Babbage Difference Engine, circa 1850.

In the pioneering days of reflection seismic processing travel time templates were used to hand migrate seismic shot records. In seismic processing substantial computing effort is consumed by programs based on the acoustic and elastic wave equation. The next two sections describe a *compute engine* for wave equations. This notion of computing is really different from our current digital computer technology. The idea is to construct an application specific computer or engine just for the wave equation. The first section describes an actual compute machine prototype, the Wave Equation Difference Engine by Bording (1995). The second

Manuscript received by the Editor September 16, 1996; revised manuscript received September 30, 1996.
[1]Institute for Geophysics, The University of Texas at Austin, 8701 North MoPac, Austin, Texas 78759-8397
The WEDE machine was built as part of Bording's Ph.D. program and many thanks are due to his dissertation committee of Roger Wainwright, Jeff Hensley, Chris Liner, Ken Kelly, Gerald Kane, Larry Lines and Sven Treitel. Julio Díaz of the University of Tulsa was the committee chair and most helpful in his guidance and support. In 1994-1995 my gracious host was Larry Lines as I was a post-doctoral fellow with the MUSIC consortium in the Earth Sciences Department at Memorial University of Newfoundland. I wish to thank the Natural Sciences and Engineering Research Council of Canada (NSERC) and Petro-Canada for their financial support of the Chair in Applied Seismology held by Dr. Lines. Thanks also to the industrial sponsors of the Memorial University Seismic Inversion Consortium (MUSIC) project for their financial support. In August, 1995 I was invited by Paul Stoffa to join the Institute for Geophysics at The University of Texas at Austin for a two year post-doctoral fellowship where support was provided by the U.S. Office of Naval Research under grant number N00014-94-1-0508.

section describes a proposed wave machine or engine for 3D pre-stack depth migration which shall be called an imaging engine.

Wave equation difference engine

The wave equation difference engine (WEDE) is a prototype parallel computing machine constructed to solve a finite difference approximation to the acoustic wave equation. The prototype has four processors which perform all the arithmetic operations in special digital hardware. This hardware design is linearly scalable in the number of processors without any loss of efficiency and overcomes many defects of existing parallel computers. The compute engine performs shot record and exploding reflector modeling. Further, because of the time symmetry of the wave equation this modeling computer can perform reverse time migrations, (RTM) (Hemon, 1978), (Loewenthal et al., 1976), (Whitmore, 1983), (McMechan, 1983), (Baysal et al., 1983), (Chang and McMechan, 1986), (Dong and McMechan, 1993), and (Bording and Liner, 1994).

The wave equation plays an important part in the processing of seismic data. Its principal use is to relate data collected on a recording surface to the interior of the earth. Waves propagating within an inhomogeneous media generate reflections which are then recorded, a seismic experiment. Using known velocity models of the earth it is possible to simulate this experiment. These synthetic experiments use the wave equation to generate seismograms with characteristics similar to field experiments (Alford et al., 1974), (Kelly et al., 1976), and (Kelly et al., 1982). The computational effort for three dimensional (3D) seismic modeling is enormous (Baker, 1989) (Holberg, 1989), and the development of wave machines can make modeling both affordable and timely. The practical use of 3D is illustrated by the pre-stack depth migrations complex salt structures of (Ratcliff et al., 1994).

A parallel implementation of a seismic depth migration using message passing and a cluster of workstations was reported by (Murillo, 1996). His results for ten processors show a reasonable efficiency for each processor of 94 per cent. The results are not as good for twenty processors where the individual processor rate drops to 50 per cent. I conclude the exchange of data is limiting the processors. His results are shown in Table 1. The expected time is the single processor time divided by the number of parallel processors in a cluster. Here the efficiency of computation is diminished as the number of processors is increased. At fifty per cent efficiency it is clearly better to use two clusters of 10 processors. Two complete jobs could be run by two clusters in essentially the same time. This raises the question, how does one go about making seismic modeling and migration work in parallel?

A prototype two dimensional wave machine has been constructed which demonstrates the feasibility of a massively parallel computational engine for seismic modeling (Bording, 1995). The elastic wave equation can also be transformed into a difference engine and this is an ongoing research project. For this paper the focus will be the acoustic wave equation. Notice the description does not use the word computer. The generality of the word computer implies the ability to solve different problems. Here the use of the descriptive term *engine*, in particular, the difference engine means a device which computes differences. The approximation used with the wave equation requires the construction of numerical differences or derivatives. Commonly used methods for solving the wave equation include Fourier, finite elements, and finite differences. Each method has advantages and limitations but all require arrays of storage and computation. These finite difference methods are well suited for parallel computation. The local nature of finite difference methods provide a distinct advantage and will be considered further. This does not preclude finite element methods or Fourier schemes from being useful as parallel methods for solving the wave equation. However, these other methods will not be considered here. Besides the local nature of finite differences, the regularity of the grid matches the structure of rectangular arrays.

Table 1. A Parallel Depth Migration, Scalable but not Linear Computing.

Processors	Elapsed Time (Seconds)	Expected Time (Seconds)	Processor Efficiency %
1	6600	6600	100
10	701	660	94
20	654	330	50

Prototype

The wave equation difference engine (WEDE) was constructed to demonstrate that a truly parallel implementation of the finite difference equation is feasible. The hand built WEDE prototype is shown in Figure 2. Most existing implementations of parallel computation use distributed processors connected via a high speed switch or network. The notable exceptions were the CM-1 and CM-2 by Thinking Machines which are no longer in production. The CM machines had special routing hardware functions and interprocessor connections. In general, the limitations of these

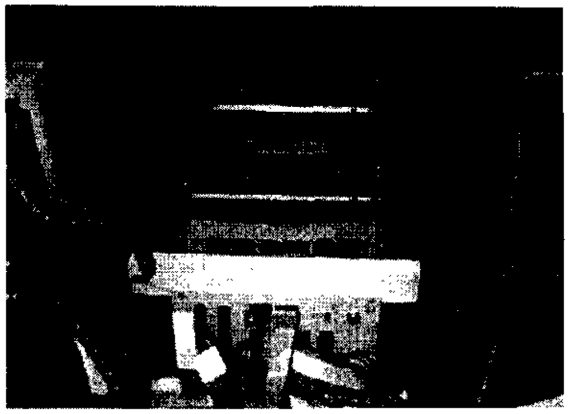


Fig. 2. Wave Equation Difference Engine by Bording.

networks become evident as more processors are used and as more data is exchanged. One way to overcome these limitations is to eliminate the generality and make the processors and interconnections much more specific and direct. This will restrict the nature of the problems which can be computed but actually provides a benefit, a feasible and scalable computer.

The WEDE implements the acoustic wave equation using four processors in parallel. Each processor computes the grid point finite difference equation in step with the neighboring processors. The method requires two independent results for each grid point evaluation. The arithmetic within each processor is computed in parallel. Thus the machine has three levels of parallel computation, the processors, the arithmetic, and the algorithmic results. The registers are designed to hold all data and single memory load is used each computation cycle. Similarly all results are stored in a single cycle. The traditional instruction cycle is used, the loading of data registers, computation, and then storing of results. Within each processor multiple results are computed and data associated with the grid point selects which is stored. This allows all computation to proceed without logical processes to delay one result in favor of another. Essential to high performance is the deterministic execution of the instruction (Brent, 1973). If any irregularity is allowed then it becomes impossible to control data movement.

The acoustic wave equation is shown in Equation 1 for a constant density media. The wave speed is C, the wave field is Ψ, and the subscripts indicate the spatial dimensions, x and z. The source function is *src(t)*.

$$\frac{\partial^2\Psi}{\partial x^2}+\frac{\partial^2\Psi}{\partial z^2}=\frac{1}{C^2}\frac{\partial^2\Psi}{\partial t^2}+src(t). \qquad (1)$$

The source is available at every grid point and applied only where selected. Each grid point has a control word which applies the source and receiver criteria. The boundary conditions are applied everywhere using damping weights (Cerjan et al., 1985) and the interior is kept constant. The damping zone is 20 or more grid points depending on model.

Model results

The limitations of the prototype restrict the size of models which can be run. The test problem consists of two regions with a flat separation layer. A synthetic model is defined by setting the geological velocity parameters on the finite difference grid. The other parameters needed are the type of source and source location. The location of the receivers and the expected frequency content of the model must also be known. These parameters specify a model; the additional information needed determines the accuracy and level of acceptable error. The number of grid points per wave length is 10. The initial model has two layers shown in the initial frame in Figure 3. The interval velocities in metres per second are (1500 *and* 1800) in sequence from the surface. This flat layer model has a reflector at a depth of 100 metres. The source is placed in the center of the model, and the source is a Ricker wavelet. A 25 frame movie was recorded during a typical run. Five of the frames are shown in Figure 3. The difference between reflecting and non-reflecting boundaries is observable in Figure 3 by comparing the top and left hand sides. In this run the damping boundary condition is applied only to the sides of the model. The reflection just starting in Frame 25 from the top demonstrates the resulting hard boundary.

Massively parallel computing

The addition of another processor to the WEDE architecture design requires additional memory and interconnections to the existing processors. These devices are connected in a linear fashion and each incremental processor operates just like all the rest. The number of processors is only limited by the size of the problem. In two dimensions assume the model size is 2,000 by 2,000 then a linear scheme would use 2,000 processors. If each processor has a 10 to 100 megaflop rate the machine performance would be 20 to 200 gigaflops. Three dimensional problems would use a two dimensional mesh. This mesh inner-connected processor would have a performance rate of 40 to 400 teraflops.

Modeling capabilities

The WEDE machine is capable of computing shot records and exploding reflector zero offset sections. If desired it is also possible to compute reverse-time migrations. This will require an additional input data space to hold the surface boundary condition data. Shot record modeling requires an input velocity field, a source location, and a set of output recording locations. The source function is excited at the source position and when the wave field passes the receiver locations it is recorded. The sources and receivers do not have to be at the surface. Hence, VSP and cross well synthetics are also possible.

The time required for a 3D shot record model is dependent on many factors. If we assume a 100 wave length cube, 10 grid points per wave length, a modeling frequency of 60 Hertz and a maximum to minimum velocity ratio of 4 then the computation effort of a shot record can be estimated. Using a square grid of 256 x 256 processors and a processor computation rate of 6 megaflops the time to compute a shot record would be 600 seconds. For a grid of 512 x 512 processors and a processor rate of 20 megaflops the shot time is 50 seconds. The approximate total engine rate is 400 gigaflops for the first case and 5 teraflops in the second.

Exploding reflector modeling is more complex and requires a reflectivity map to adjust the source strength. Each location in the model where the impedance change is sufficient to cause a reflection is considered a source point. All the sources go off at once, and using half of the model velocity waves are generated which propagate in all directions. The data are recorded in a manner similar to shot modeling. Reverse-time migration needs only the seismic time data placed at the recording surface. This is possible by using an

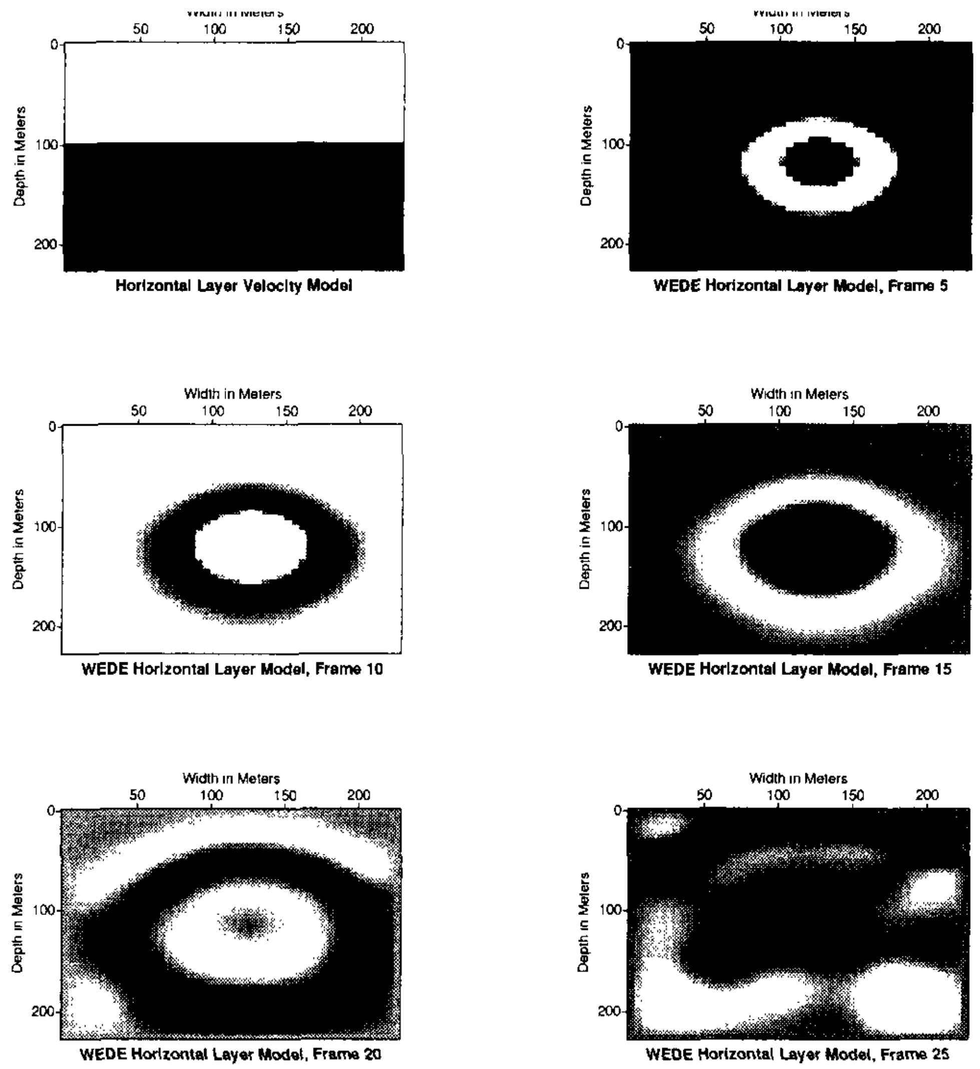


Fig. 3. Model and Movie Frame Snap Shots.

additional data field. Initially the seismic data are transferred to an array and at each time step the data are applied as a boundary condition at the recording surface.

Summary

The WEDE prototype demonstrates that the notion of an application specific wave equation computer is feasible. It is now possible to build large scalable parallel wave equation engines for seismic shot record modeling. The existing difficulties of existing parallel computers for speedup can be overcome by explicit communication paths and the other unique features of the WEDE. In the zero offset domain these computational engines can be used for exploding reflector modeling and reverse time migrations.

Introduction to Imaging Machines

The Imaging Wave Machine (IWM) is proposed for three dimensional pre-stack depth migration (Bording, 1995a). The computational intensity of seismic migration and inversion limits the processing of three dimensional data. The processing of two dimensional data illustrates the robustness of the applied mathematical tools and the current level of computational resources for imaging the earth's subsurface. The additional effort required by three dimensional data is several orders of magnitude more and results in only the simplest of mathematical algorithms being used on the most powerful computers available today. This leads to a costly solution with a less desirable algorithm; clearly an unacceptable situation.

A solution to the computational resource problem can be found in the nature of seismic data. A high level of parallelism can be found in some of the wave equation based algorithms. By exploiting these parallelisms and using new computer architectural methods it is possible to construct wave engines for seismic inversion in three dimensions. These new methods use multiple parallel independent memories and a processor architecture specific to the wave equation.

As discussed above the basis for the IWM is my recent dissertation where the prototype seismic modeling wave machine is presented. The suggested seismic method for pre-stack reverse-time migration (PRTM) is based on the wave equation and overcomes the dip limitations of lesser methods. The PRTM method uses the explicit finite difference formulation of the wave equation and processes the seismic data as the surface boundary condition.

Teraflops

A Teraflop is a million million floating point operations, about 100,000 times faster than your typical personal computer. The future of high performance computing, Teraflop and above, will require a significant number of processors working together. The organization of these processors will fall into two broad categories. Massively parallel machines or synchronous processors which will compute regular instruction streams and asynchronous processors which will compute regular and/or irregular instruction streams. The organization of existing architectures will adapt to the algorithmic needs of the applications. These adaptations might result in revolutionary or evolutionary designs. But whatever changes take place in the computer architecture the usefulness of the applications operating on these machines will determine criteria for success or failure.

A massively parallel computer is proposed which solves a class of partial differential equations using finite difference methods. The application proposed here is the three dimensional pre-stack imaging of seismic data. The machine is designed to use three dimensional arrays of data which pass through a grid of processors. Due to the local nature of the finite difference method it is possible to arrange the data movement to achieve high individual element processor instruction rates. Using many, a million, of these complex instruction processors, it is possible to operate at rates above a teraflop. Software and language requirements are limited because much of the instruction processes are explicit. The host environment is restricted to loading initial data and recovering results.

Megaflops, gigaflops, and all the flops

The current measure of performance for computing is millions of floating point operations per second, the Megaflop. To achieve a megaflop, the memory systems must operate at several million load/store operations per second. Arithmetic processors are capable of performing the four basic operations of addition, subtraction, multiplication, and division within one microsecond. This level of performance is now available from single chip VLSI microprocessors.

Two different routes have been taken to operate at arithmetic rates in excess of a megaflop. The first is to increase the clock speed of the single chip processor. The second is to construct pipeline arithmetic units which require a startup or initial load process. The time for a single result is an integer multiple of the pipeline clock. After initial startup a result is obtained for every clock. Combinations of pipelines are also used in the single chip VLSI processors. In fact, gate densities are now such that a single chip can have several arithmetic cells.

Reaching gigaflop performance, (a thousand times the megaflop), still requires the assembly of several arithmetic processors. Therefore, the control and interfacing of memory, registers, and the arithmetic units becomes a significant part of the design. In two-dimensional computational algorithms it is possible to construct methods which can use a thousand processors effectively. The success of the CM-2, which used 1024 floating point arithmetic processors to achieve 10-30 gigaflops, is an example of this approach.

The construction of teraflop machines using current designs will be based on increasing the number of processors by a factor of ten and increasing the clock rates to effective rates above one hundred megahertz. The processor count is on the order of ten thousand and the expected processor arithmetic rate is one hundred megaflops per arithmetic unit. This assembly of processors requires a large and general purpose interconnection network to route data as needed by the different computational algorithms.

Current limitations

In the above description of current machines nothing is said about real efficiency, about the details of the processors or enhancements which might be used to expand capability. Certainly, enhancements can be made to existing processors and interconnection schemes and will be made. The underlying trends of higher density VLSI processors, improvements in programming languages, and the much needed improvements in interconnection topology and speed will lead to faster machines.

Of these trends let us consider the programming language trend. If Fortran continues as the development tool for scientific computing who will provide the necessary support for the user community? Where will the funds come from for the software support for the continuing development of Fortran compilers? The Connection Machine compiler reached maturity shortly before the demise of the company and long after the machine development cycle was complete. The ETA project was terminated before the Fortran compiler was useful. Of all the limitations for using high performance computing the software bottleneck is the most punishing. It must be addressed if the machines are to be useful within a reasonable time after construction.

The hardware trends are much more difficult to predict, but the most awkward to manage is the growing gap between memory cycle times and the processor speed (Stallings, 1993). The two basic kinds of memory are static and dynamic, static uses an active transistor storage cell and dynamic memory uses a capacitor to store a charge. Each has its merits, static is larger and faster, and dynamic is slower but much more compact. For very large machines the power consumption favors dynamic random access memory (DRAM). Static random access memory has the distinction of simplicity of operation. In either case, the issue of data movement to and from memory must be addressed by new designs. Existing processors now operate at clock rates which are faster than the memory access times. Efficiency

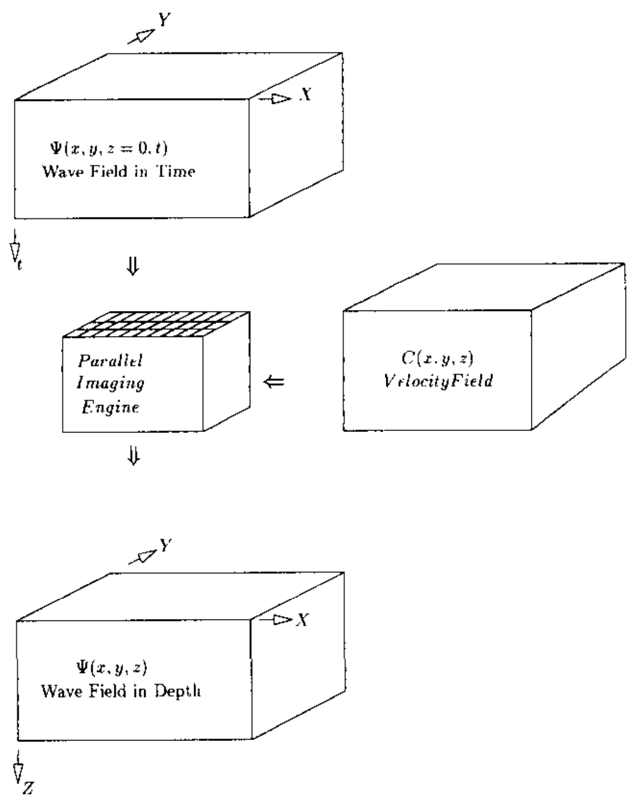


Fig. 4. The Imaging Wave Machine, Post-Stack.

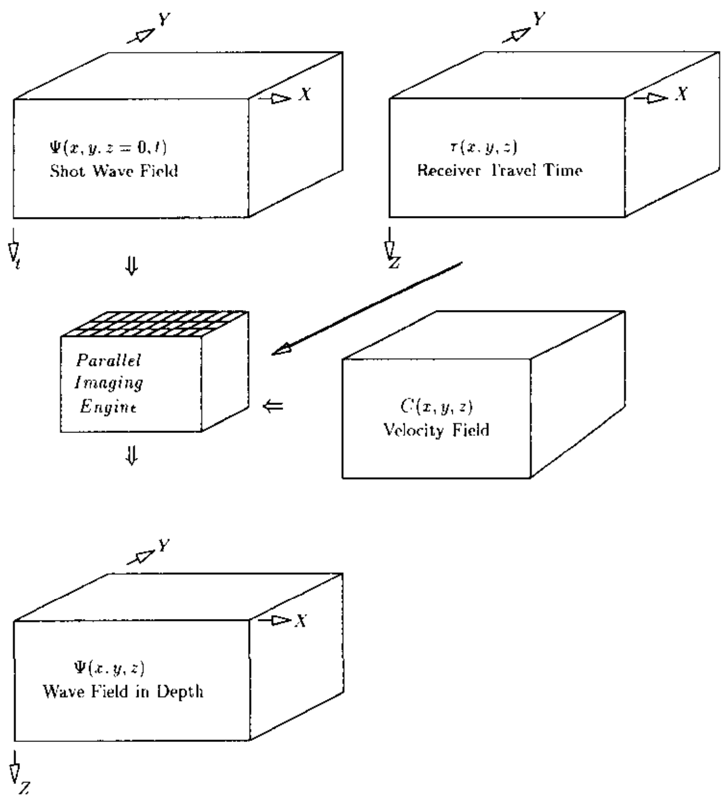


Fig. 5. The Imaging Wave Machine, Pre-Stack.

then becomes problem dependent, if the algorithm has high data reuse and locality then the processor will have sufficient data for high instruction rates. Otherwise, the processor will have to wait for data from memory and from other processors and have significantly lower instruction rates. The different approach used here had the memory system designed to provide each processor with all necessary data for the execution of each complex instruction.

Non-traditional architectures

The current register instruction sets are defined using single or at most two arguments and generate a single result. This places a burden upon the processor to present all the data sequentially. Some newer processor designs examine the incoming instruction stream and execute instructions out of sequence if analysis determines that they are independent of those currently executing. This hardware analysis is expensive in terms of chip floor space and only achieves a modest improvement. A better approach would be to determine from the problem the nature of the instruction stream and design a processor capable of efficiently managing specific problems. For example, assume a program computes the Fourier transform many times during a single run. Why not have an instruction just for the transform operation? By placing the data into a special register and performing the necessary operations the program is faster and can make good use of the specialized hardware (Brent, 1973). No compiler assistance is needed other than calling the hardware device.

The basic method of memory-register-processor used in current architectures is limited in the sense that no implicit sharing of data is allowed between processors. All data must be explicitly moved prior to sharing. This can be a very time consuming process and must be overcome if processors are to be kept busy. In existing memory designs the use of intermediate memories which are smaller but faster help improve the rate at which data can be accessed. These cache storage

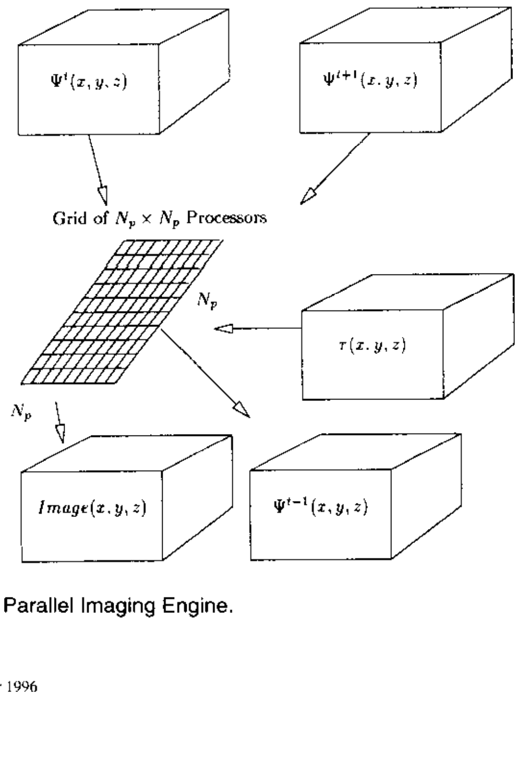


Fig. 6. Parallel Imaging Engine.

3D Migration Scale

- N_x is the number of grid points in the X Direction, $N_x \cong 1024$.
- N_y is the number of grid points in the Y Direction, $N_y \cong 1024$.
- N_z is the number of grid points in the Z Direction, $N_z \cong 1024$.
- N_t is the number of time steps required for the simulation, $N_t \cong 2048$.
- w/s is the number of arithmetic operations per grid point, $w/s \cong 64$.

The memory requirement of eight spatial cubes $(8 \times N_x \times N_y \times N_z)$ and two data cubes, $(2 \times N_x \times N_y \times N_t)$ is;

$$M(N_x, N_y, N_z, N_t) \cong 1.2 \times 10^{10} \text{ memory words.}$$

The computational effort is; $O(N_x, N_y, N_z, N_t, w/s) \cong 1.3 \times 10^{14}$.

Total Operations/Shot	Rate x Processors	Time	Entire Pre-Stack 320 x 320 Shots
10^{14}	Parallel $160 \times 10^6 \times 1024 \times 1024$	1 Second	1 Day

Fig. 7. Complexity of 3D Pre-stack Migration.

devices introduce great irregularity to expected completion times and make it difficult to maintain synchronous performance.

The rate differences between various parts of the system, memory to registers and registers to processor function units, and then registers back to memory play a significant part in making these designs difficult. In cache systems it is essential to maintain correct data within the cache and into the main memories. The IWM independent parallel memory organization eliminates these design and execution difficulties.

Massively parallel and 3D

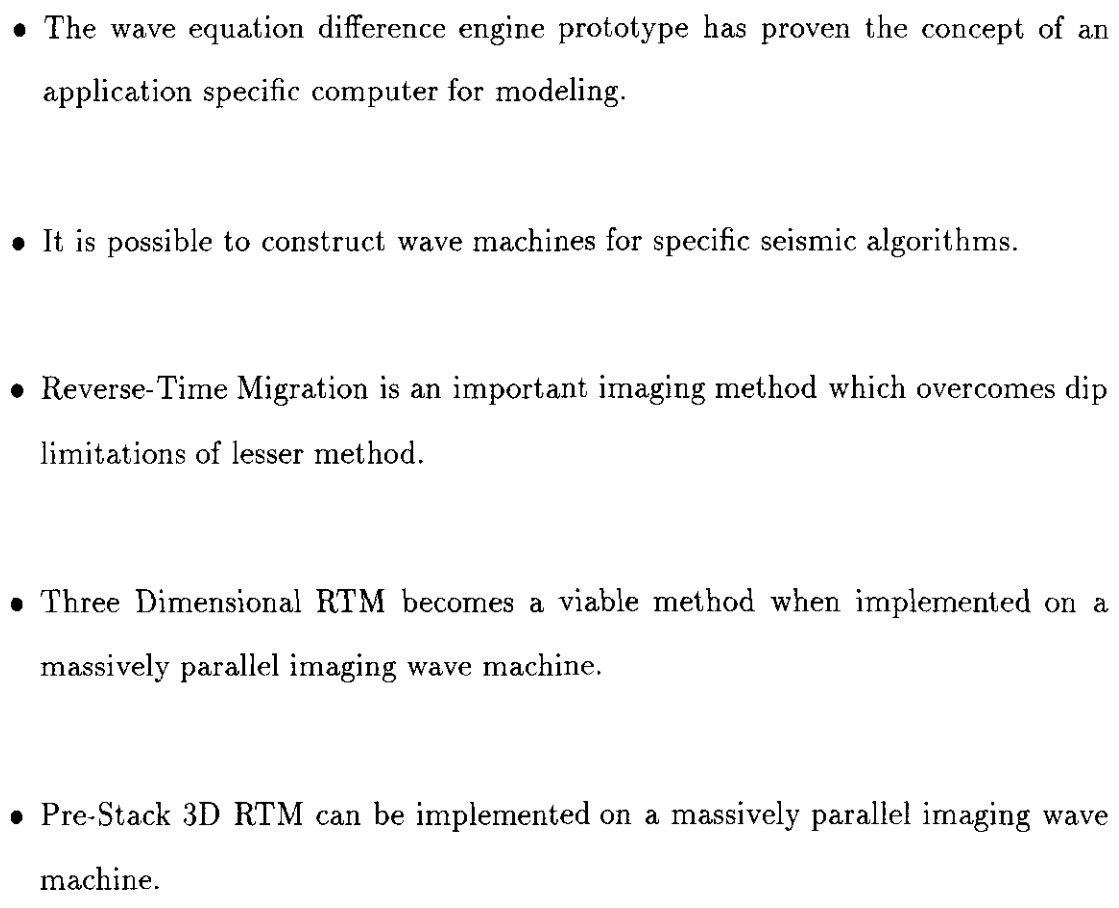
The imaging of seismic data is computationally intensive as described in several papers by (Mufti, 1989), (Mufti, 1990), (Mufti et al., 1996), and (Wu et al., 1996a), (Wu et al., 1996b). A massively parallel wave equation computer is considered here as a computing resource for seismic imaging. Reverse-time migration for zero-offset data is a suitable method for parallel computation. The data array movement for post-stack three dimensional reverse-time migration is shown in Figure 4. The prototype wave equation difference engine is extended to more processors, to a higher order difference approximation, and to different types of boundary conditions. The finite difference wave equation approximation has the unique feature of being able to run backwards in time. This allows the surface seismic data to be introduced as a boundary condition and, as time is reversed, generate a subsurface image.

The computational merit of reverse time imaging in 3D is the simplicity of the finite difference method, which requires three spatial dimensions and time. Other methods, Kirchhoff for example, require the construction of travel time tables and an imaging condition which could require as much as another order of magnitude of processing.

Using the acoustic wave equation, three dimensional pre-stack reverse time migration propagates the recorded wave field backwards in time and correlates it with the source travel-time. The intermediate results are summed to construct an image of the subsurface. This pre-stack process is illustrated in Figure 5. For elastic data a similar algorithm is possible but the complexity of the staggered grid methods are increased by a factor of ten or more.

The grid point template (the difference molecule) requires the evaluation of derivatives at each spatial location. These local operations, the inner products of the data with the difference weights, and the time differencing are computed directly by the processor. This reduces the number of instructions cycles needed for the arithmetic to one. The parallel operation along area of subset of the area of two of the dimensions allows the other significant efficiency improvement. The interconnection of processors is unique in the wave equation difference engine (WEDE) design and allows each processor access to all needed data for each instruction cycle. No queues, no message passing, and no delay is allowed within the design. These constraints overcome much of the difficulty with current parallel designs, and the essential data movement is shown in Figure 6.

This is an application specific computer which solves a class of time marching algorithms for hyperbolic partial differential equations. The WEDE used a second order in time and second order in space finite difference algorithm. Here the spatial differences are extended to fourth and beyond to improve accuracy. The boundary conditions are extended to allow the surface seismic data to be properly introduced. The



- The wave equation difference engine prototype has proven the concept of an application specific computer for modeling.
- It is possible to construct wave machines for specific seismic algorithms.
- Reverse-Time Migration is an important imaging method which overcomes dip limitations of lesser method.
- Three Dimensional RTM becomes a viable method when implemented on a massively parallel imaging wave machine.
- Pre-Stack 3D RTM can be implemented on a massively parallel imaging wave machine.

Fig. 8. Summary.

WEDE processor interconnection scheme is extended by adding the additional third dimension but this does not require any significant change in design.

The entire array of processors is controlled by a state machine which is started by a host processor. Upon completion of the initial data insertion, the engine is started and needs no intervention to complete its task. The velocity field and the surface data are provided internal storage arrays. All computational data storage is allowed for in the processor memories. This overcomes the Connection Machine handicap of a slow control processor. As the processors operate, an output facility will provide an independent, asynchronous data transfer for visualization processing. Upon completion of the run the resulting image can be moved to a seismic interpretation processor. This will facilitate the extensive man-machine analysis that is required to interpret the 3D data volumes.

Assuming a grid of 1024 by 1024 processors and associated memory arrays the processors could execute 64 arithmetic operations each instruction cycle. All the computations associated with a grid point number 64. The grid point processors will complete all computation in a memory cycle time giving an effective instruction rate of several hundred megaflops at todays memory cycle times. The size of a typical inversion data volume is shown in Figure 7 along with the operation counts and surface data size. Assuming 320 by 320 shots, a complete pre-stack data migration will take one day at a 160 teraflops computation rate.

Summary

The main points of this paper are shown in Figure 8. Imaging three dimensional seismic data is computationally intensive and the use of wave equation difference engines can reduce the time required to pre-stack migrate large data sets. The WEDE prototype provides a basis to build finite difference engines for migration. These massively parallel machines could have as many as a million processors, and would be capable of completing an entire 3D pre-stack migration in one day. I predict that application specific computing will become the method of choice for seismic processing in the future.

References

Alford, R.M., Kelly, K.R. and Boore, D.M., 1974, Accuracy of finite-difference modeling of the acoustic wave equation: Geophysics **39**, 834-842.

Baker, L., 1989, Is 3-d wave-equation modeling feasible in the next ten years, *in* Eisner, E., Ed., Supercomputers in Seismic Exploration: Pergamon Press.

Baysal, E., Kosloff, D. and Sherwood, J., 1983, Reverse-time migration: Geophysics **48**, 1514-1524.

Bording, R.P. and Liner, C.L., 1994, Theory of 2.5d reverse time migration: SEG Annual Exposition Expanded Abstracts, 692-694.

________, 1995a, Pre-stack depth migration in three dimensions with imaging wave machine: SEG Annual Exposition Expanded Abstracts, 1005-1008.

Bording, R.P. 1995, Wave equation difference engine: Ph.D. thesis, The University of Tulsa, Tulsa.

Brent, R.P., 1973, The parallel evaluation of arithmetic expressions in logarithmic time: Academic Press, New York, New York.

Cerjan, C., Kosloff, D., Kosloff, R. and Reshef, M., 1985, A nonreflecting boundary condition for discrete acoustic and elastic wave equations: Geophysics **50**, 705-708.

Chang, W.F. and McMechan, G.A., 1986, Reverse-time migration of offset vertical seismic profiling data using the excitation-time imaging condition: Geophysics **51**, 67-84.

Dong, W. and McMechan, G.A., 1993, 3-d prestack migration in anisotropic media: Geophysics **58**, 79-90.

Hemon, C., 1978, Equations d'onde et modeles: Geophys. Prosp., **26**, 790-821.

Holberg, O., 1989, Wave equation computations and truly parallel computations, *in* Eisner, E., Ed., Supercomputers in Seismic Exploration: Pergamon Press.

Kelly, K.R., Ward, R.W., Treitel, S. and Alford, R.M., 1976, Synthetic seismograms: A finite difference approach: Geophysics **41**, 2-27.

________, Alford, R.M. and Whitmore, N.D., 1982, Modeling – the forward method, *in* Jain, K.C. and deFigueiredo, R.J.P., Eds., Concepts and Techniques in Oil and Gas Exploration: Society of Exploration Geophysicsts.

Loewenthal, D., Lu, L., Roberson, R. and Sherwood, J., 1976, The wave equation applied to migration: Geophys. Prosp. **24**, 380-399.

McMechan, G., 1983, Migration by extrapolation of time-dependent boundary value: Geophys. Prosp. **31**, 413-430.

Moseley, M., 1964, Irascible genius, a life a Charles Babbage, Inventor: Hutchinson, London.

Mufti, I.R., Pita, J.A. and Huntley, R.W., 1996, Finite-difference depth migration of exploration-scale 3-d seismic data: Geophysics **61**, 776-794.

______, 1989, Application of supercomputers in three-dimensional seismic modeling: Pergamon Press, Oxford.

______, 1990, Large scale three dimensional seismic models and their interpretive significance: Geophysics **55**, 1166-1182.

Murillo, A.E., 1996, A parallel implementation of a depth migration code: Computational Science Education Project, http://ppl.mines.colorado.edu/SeisMigration/SeisMigration.html.

Ratcliff, D.W., Jacewitz, C.A. and Gray, S.H., 1994, Subsalt imaging via target oriented 3-d prestack depth migration: The Leading Edge **13**, 163-170.

Stallings, W., 1993, Computer organization and architecture: Macmillan, New York, New York.

Whitmore, N.D., 1983, Interactive depth migration by backward time propagation: SEG, 53rd annual meeting in Las Vegas, Nevada.

Wu, W., Lines, L.R., Burton, A., Lu, H., Jamison, W., Zhu, J. and Bording, R.P., 1996a, Prestack depth migration of an Alberta foothills data set: CSEG, 1996 annual meeting in Calgary, Alberta.

Wu, W., Lines, L.R. and Lu, H., 1996b, Analysis of higher-order finite-difference schemes in 3-d reverse time migration: Geophysics **61**, 845-856.