Wave Inversion Technology, Report No. 2, pages 261-269

Distributed 2D Seismic Modeling and Processing

M. Karrenbach[1]

keywords: *parallel, processing, modeling*

ABSTRACT

In Geophysics measured seismic data sets need to be analyzed using numerical operators. Seismic data can be forward modeled from hypothetical earth models with high levels of complexity. Both steps are necessary ingredients for seismic data inversion procedures. The result of such inversion methods is a realistic description of the subsurface of the earth as derived from measured data. I show a distributed version of a seismic utility package, called SEPLIB, that can be used in a parallel computer environment such as the IBM SP-2. Modeling and processing applications for 2D seismic data are shown, that use a high-level data and process distribution mechanism, called Sepzilla and Parmod2d. This mechanism achieves coarse grain parallelism for 2D seismic problems.

INTRODUCTION

The memory and time requirements of seismic calculations suggest parallel implementations. Geophysical experiments usually produce large data sets (Gigabytes and Terabytes), when structure and material properties are to be analyzed in three dimensions. Modeling of seismic data usually starts with an input earth model that is only a few Gibgabytes of size, but on output it can produce as large data volumes as can be found when carrying out measurements in reality. Seismic processing uses large data volumes to extract information about the earth's material parameters. The next two sections describe parallel application of two-dimensional seismic processing and modeling algorithms.

SEISMIC DATA PROCESSING

Analysis of those data applies sequences of mathematical operations. Often these operations can be described by linear and non-linear operators, such as Fourier Transforms,

[1]**email:** martin.karrenbach@physik.uni-karlsruhe.de

convolutions and other integral or differential transforms. In the end, the seismic data inversion produces a detailed description of the earth's interior. Figure 2 shows the result of a two-dimensional acquisition geometry for a real experiment (Courtesy of Mobil Oil Co., USA). The three faces of the three-dimensional data volume are projected on to two dimensions. The gray-scale amplitudes depict the signal strengths of reflected and recorded signals from the subsurface. Several high contrast layers give rise to strong reflection amplitudes.

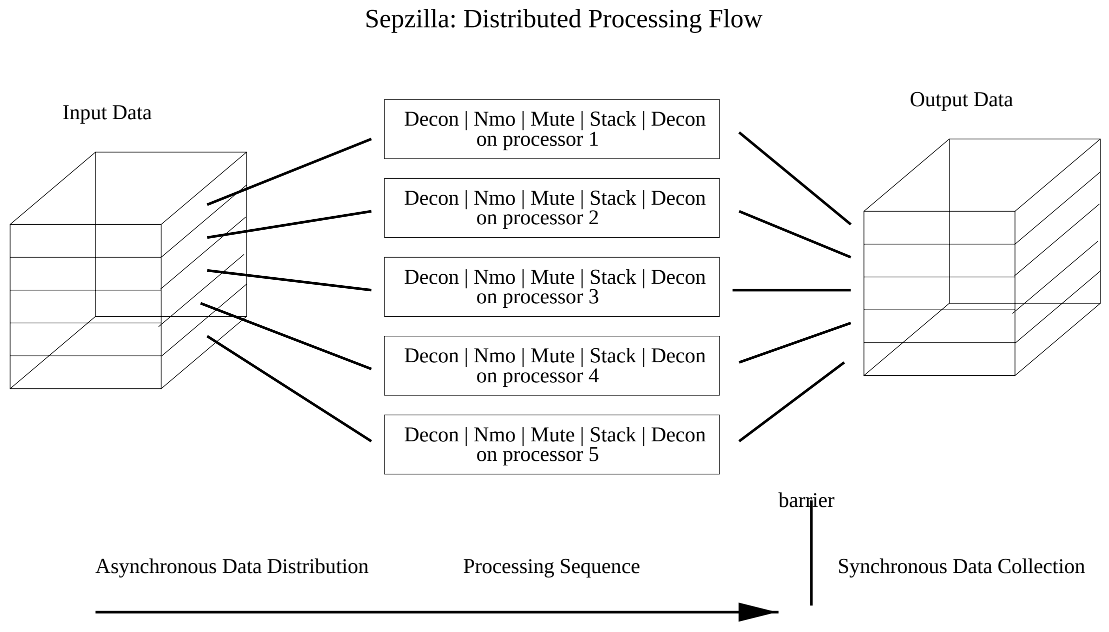
Sepzilla: Distributed Processing Flow
Input Data
Decon | Nmo | Mute | Stack | Decon on processor 1
Decon | Nmo | Mute | Stack | Decon on processor 2
Decon | Nmo | Mute | Stack | Decon on processor 3
Decon | Nmo | Mute | Stack | Decon on processor 4
Decon | Nmo | Mute | Stack | Decon on processor 5
Output Data
barrier
Asynchronous Data Distribution
Processing Sequence
Synchronous Data Collection

Figure 1: A seismic input data set is split along one of its axes. Data subsets and processing sequences are distributed on various nodes asynchronously. Results are collected after all processes are finished and the final output is assembled.

The Sepzilla data distributor splits the input data set across a user defined axis (often the third dimension). The data and process distribution is achieved by using the DCE remote shell `dce_rsh`. A suitable number of remote processes is started initiating the user-defined processing sequence on a remote node. The remote processes can retrieve the input data segments either via Socket communication or DFS file access. The file access method is the preferred method if all processing nodes see the identical file system spaces. This requirement is fulfilled on the IBM SP-2 with its DFS and Parallel I/O file system. Sepzilla is then monitoring the processes for the finish of computation. The remote processes typically write output data to local files which are combined after the last process has finished. Alternatively output data could be retrieved via a socket communication mechanism, but nearly always the local file collection proved to be speedier. Sepzilla follows the Single Program Multiple Data methodology. The Single Program is in fact a single sequence of piped processes that is applied to a data subset. Load balancing is then automatically achieved by properly partitioning the input and output data spaces.

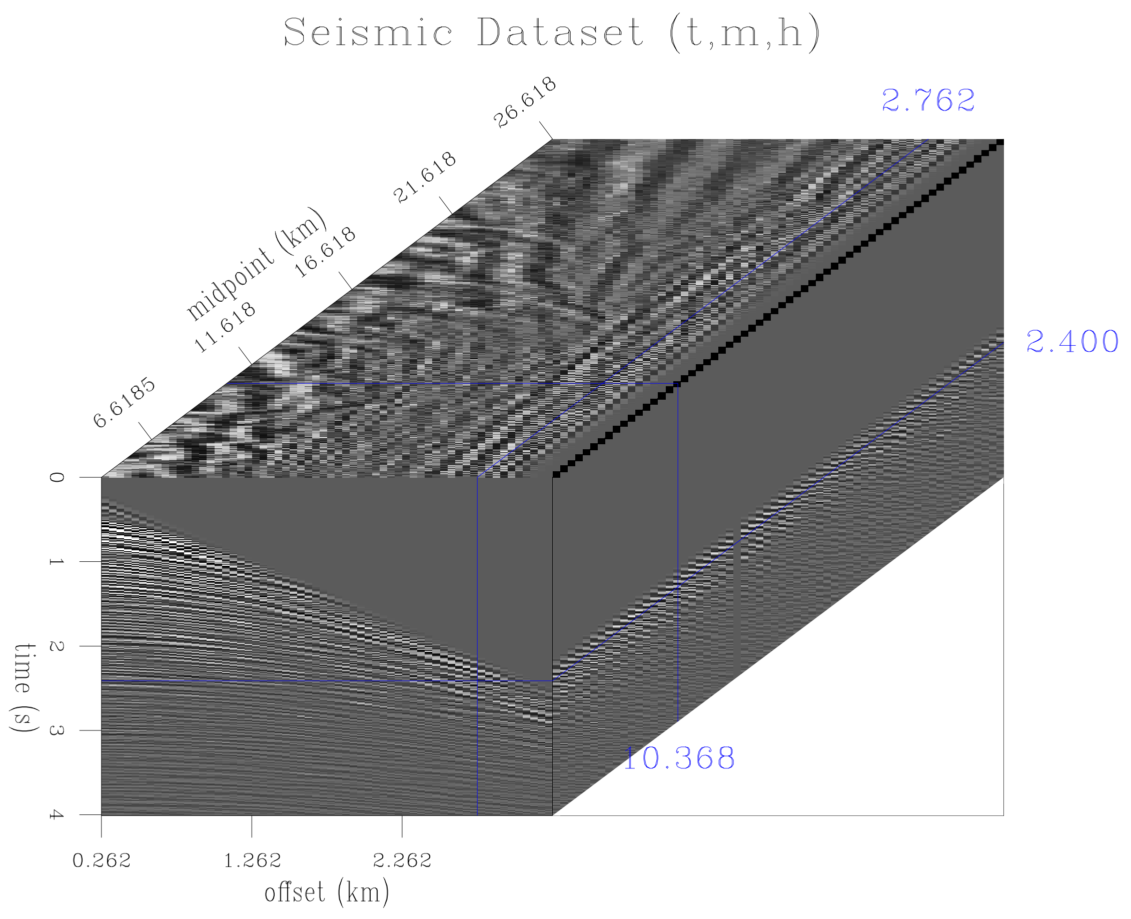

Figure 2: Input seismic data cube courtesy of Mobil Oil Co.

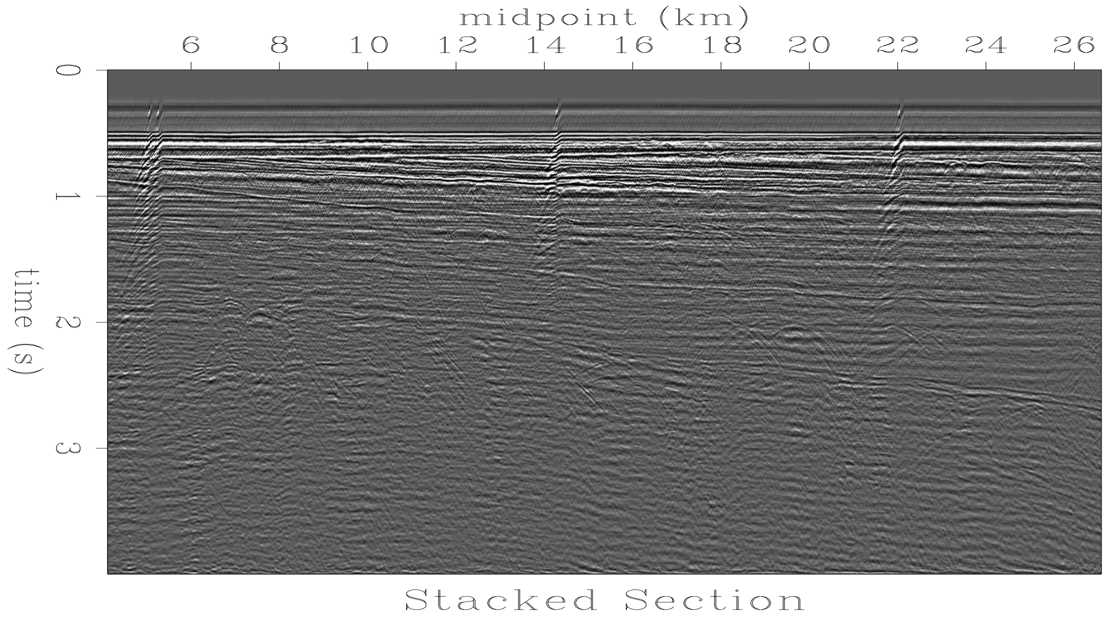

Figure 3: Output seismic data section after processing.

A second data and process distribution `Pipe` works in such a way that for a sequence of processes each single process is distributed on a individual node. Then the input data is not subdivided but flows as a whole through each of the nodes. This method of operation proofed to be too costly in data communication even with the SP-2 high speed switch. Furthermore is the behavior of the sequence determined entirely by the weakest link in the chain. If a single process requires a large subset of the partially processed input data, it will block the chain until it has received all necessary data. Only after that event the process flows again freely. A typical number of nodes is 3 to 10 and a minimum of 10 MB of seismic data should be used for small seismic processing tests. Due to these deficiencies we do not use this mode of operation.

In a related project coarse and fine grain parallel operators have been implemented in the novel programming language Java. Java allows true remote object distribution by using JavaParty – a Java software add on that provides tools to handle remote threads. Some Java compilers can produce true native executable code, which achieves performance comparable to HPF programs. Details about the results have been reported in journals Jacob et al. (1998) and at various conferences. Those tests were carried out not only on the SP-2, but on a multitude of shared- and distributed-memory computers. This work was supported by various international super-computing centers and was carried out in cooperation with companies such as SUN Microsystems, Silicon Graphics Inc. and IBM Research Labs.

Seismic Data Modeling

Two-dimensional seismic modeling can be carried out in parallel. The most simple strategy is to assign to each compute node a particular set of source points for which the seismic wave field needs to be modeled. The subsurface model and the wavelet is usually identical for all source points. Figure 4 illustrates the computing concept. Since two-dimensional models usually fit entirely within the compute node memory, costly inter-node communication is saved. For realistic three-dimensional models such a simple concept is usually not applicable.

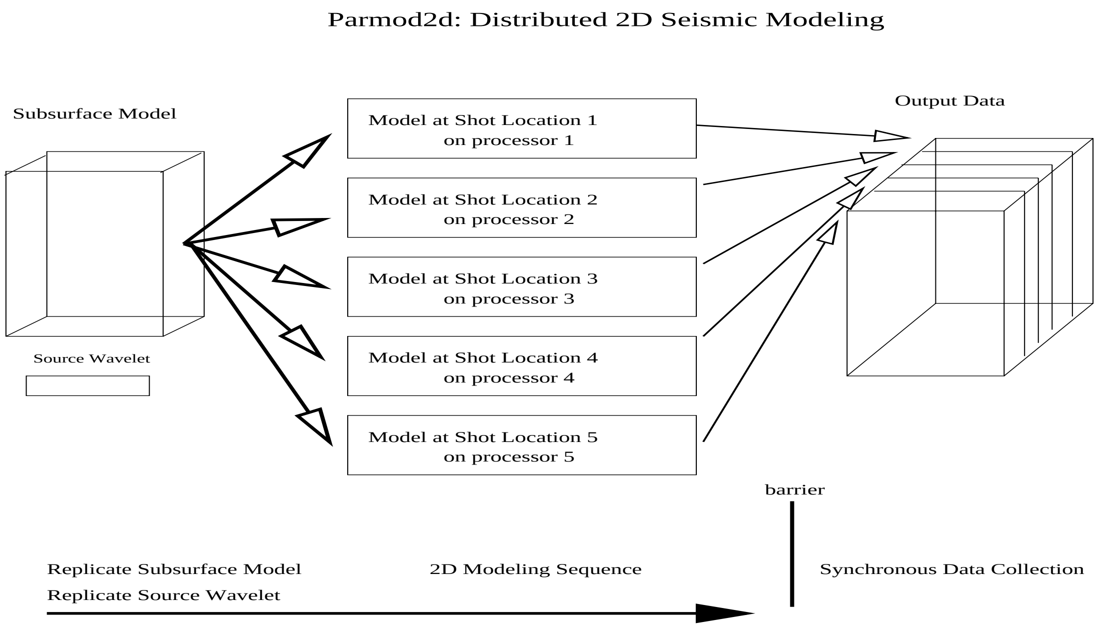
Parmod2d: Distributed 2D Seismic Modeling
Subsurface Model
Source Wavelet
Model at Shot Location 1 on processor 1
Model at Shot Location 2 on processor 2
Model at Shot Location 3 on processor 3
Model at Shot Location 4 on processor 4
Model at Shot Location 5 on processor 5
Output Data
barrier
Replicate Subsurface Model
Replicate Source Wavelet
2D Modeling Sequence
Synchronous Data Collection

Figure 4: Parallel computation scheme for 2D seismic modeling.

Parmod2D is parallel Finite Difference modeling package, that exhibits nearly perfect scalability for two-dimensional modeling. The algorithm is suited to calculate entire acoustic, elastic and anisotropic prestack seismic data volumes. Communication is only necessary when the input model and source wavelet are distributed to the compute node and when results (seismograms and snapshots) are combined into a central output file. During computation no communication is necessary. The only communication bottleneck is at the beginning, where model and source wavelet are distributed to the compute nodes and also at the end of the computation, where the resulting wave field seismograms are collected into one output file for further processing. Thus we reach nearly perfect scalability, due to the excellent load balancing.

Using this computational method we aim at generating seismic 2D benchmark data for understanding wave propagation phenomena, testing newly developed processing techniques, and finally use them for interpreting data pattern and events found in real seismic data. This is similar to the original 2D Marmousi Project or the more recent

3D modeling initiative by SEG and EAGE. In all those cases purely acoustic models were used due to the limitations in compute power and available computation time. I am going beyond the acoustic restriction in providing acoustic, elastic and more complicated media type. Leaving the structural model identical during those simulations, ensure that we get pure information about the influence of medium type choices. It is important to model seismic data more realistically such that processing methods can be truly tested with data that do not fulfill the underlying assumptions of the processing algorithm – as is the case in the earth.

I am using a 2D slice take out of a 3D model, which has been generated before as the official SEG/EAGE salt model. Figure 5 shows the structure of the 2D model. A salt lens is dominating the geology with a high velocity contrast to the surrounding material. The background velocity is smooth and given by gradual increases. Structural complexity is added by including reflecting layers and faults. Varying source points along the entire model on each discrete grid node produces a conventional fixed spread seismic survey over this model. Figure 6 shows the entire prestack seismic data set as a three dimensional rectangle, indicating time, receiver and shot axis.

The top panel shows a time slice cut through the data set a constant time. Amplitude variations indicate the presence of the salt body with its rugged topography. The front panel shows a common shot gather while the side panel displays a common receiver gather.

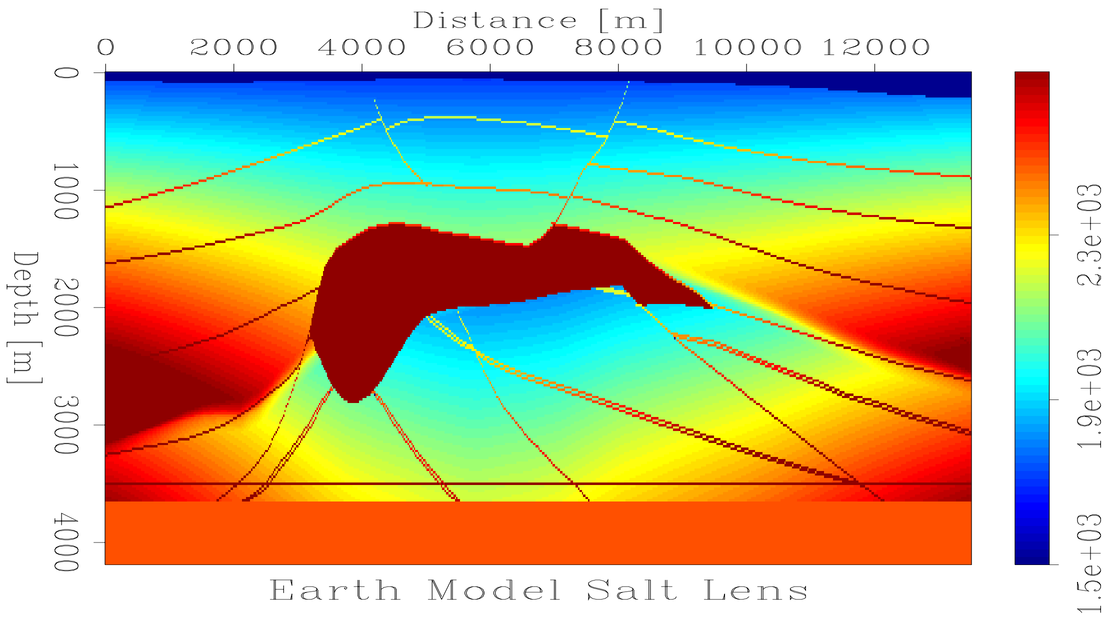

Figure 5: Subsurface input model: SEG/EAGE Salt.

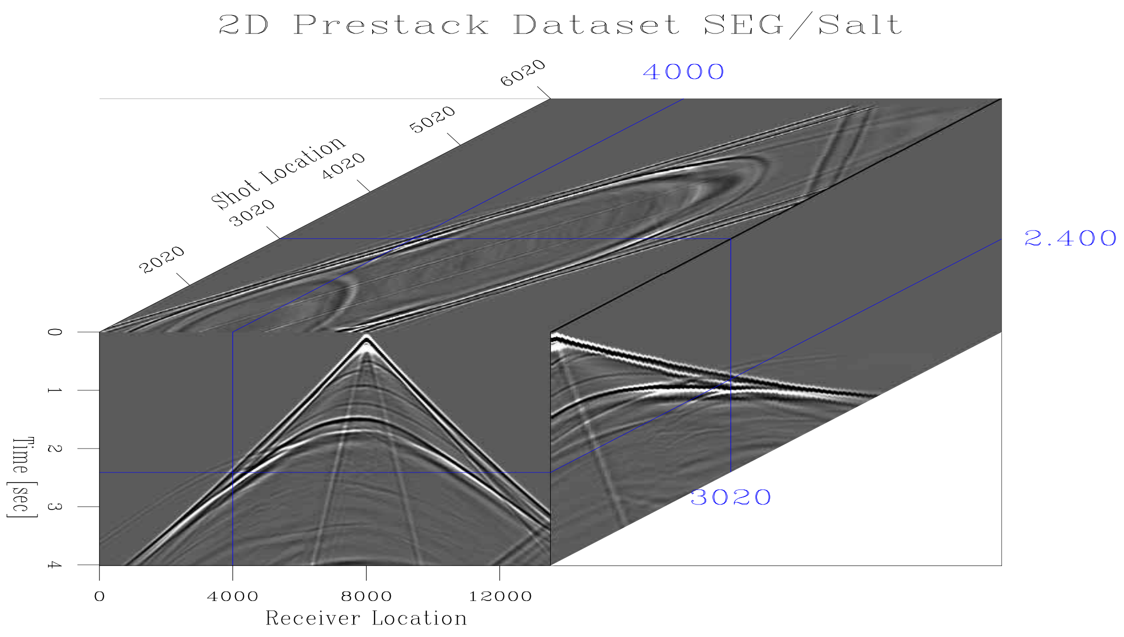

Figure 6: Example of an entire 2D prestack data volume generated from the SEG/EAGE salt model: acoustic / elastic/ with / without freesurface.

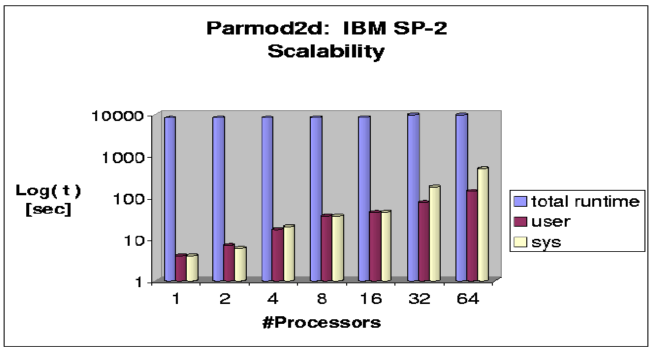

Figure 7: Runtime measurements for a varying amount of parallel processing nodes. Nearly perfect scalability of the total run time. Note the logarithmic scale of the timing measurements. Computational Time dominates the problem, communication times as indicated by `user` and `sys` times play only a marginal role.

DISCUSSION

Parmod2D is a practical software tool to model two-dimensional seismic data using a Finite Difference technique. Figure 7 shows runtime measurements for a varying amount of parallel processing nodes. Nearly perfect scalability of the total run time is achieved. Note the logarithmic scale of the timing measurements. Computational time dominates the problem, communication times as indicated by `user` and `sys times` play only a marginal role. Realistically it means, one 2D shot gather is computed in nearly the same time as 1000 2D shot gathers provided the processors are available.

Sepzilla is a tool to distribute common seismic processing sequences that can be command line specified, onto parallel compute nodes. Due to its simple nature only coarse grain parallel computational methods can be distributed easily. Fortunately standard seismic processing steps can in many cases be applied to distinct subsets of the input or output data set. In such cases Sepzilla can be used for parallelizing the operation on distributed-memory machines. It makes use of the already existing serial SEPLIB software modules that implement standard processing steps.

REFERENCES

Jacob, M., Philippsen, M., and Karrenbach, M., 1998, Large-scale parallel geophysical algorithms in java – a feasibility study: Concurrency: Practice & Experience.

Karrenbach, M., 1995, Elastic tensor wave fields: Ph.D. thesis, Stanford University.

SEPLIB. Processing package:. Stanford Exploration Project, http://sepwww.stanford.edu/, 1998.

Yilmaz, O., 1987, Seismic data processing: Society of Exploration Geophysicists, http://www.seg.org/.