Parallel space-time migration

Stewart A. Levin

Introduction

Finite difference modeling naturally lends itself to parallel processing. The simple 15° case in Figure 1 illustrates the salient features. The input subsurface model is placed on the diagonal A_i and is successively advanced towards larger time and smaller depth using a differencing star, pictured below, that computes a new vector from three previous vectors. In this way we sweep through subdiagonals B_i, C_i, etc. until the time section A_0, B_0, C_0, etc. at the surface $z=0$ is reached.

15° Finite difference modeling grid

$$
\begin{array}{llllll}
t & A_0 & & & & \\
\downarrow & B_0 & A_1 & & zero & \\
 & C_0 & B_1 & A_2 & & \\
 & D_0 & C_1 & B_2 & A_3 & \\
 & E_0 & D_1 & C_2 & B_3 & A_4 \\
 & & & & & z \rightarrow
\end{array}
$$

Differencing star

$$
\begin{array}{ll}
Y_{i-1} & X_i \\
Z_{i-1} & Y_i
\end{array}
$$

FIG. 1. Diagram of 15° finite difference modeling. Each box represents a vector of spatial data values, with the input model A_i on the diagonal and the output time section in the first column. Extrapolation is accomplished by repeated application of the differencing star which generates Z_{i-1} from Y_{i-1}, Y_i and X_i.

For us the important features of the above algorithm are that

1) the vectors on each subdiagonal are independent of each other, and
2) computations are local, involving only a few (i.e., three) input vectors to the differencing star.

This makes the computation suitable for parallel processing. Each processing element implements a differencing star and its output is used to compute the next subdiagonal. For 15° modeling we have one processing element dropping out at each stage, with the task completed when there are no processors left.

Reverse time migration, a recent development, employs a modeling program to accomplish migration. In essence, one reverses the steps in the above modeling scheme and successively computes E_i, D_i, etc. until the migrated image plane A_i is reached. Now the differencing star outputs X_i, and at each successive subdiagonal one additional processor comes on line. In this paper I will discuss generalizations and practical details of this approach.

Derivations

As always, I start from the acoustic wave equation

$$P_{xx} + P_{zz} = \frac{1}{v^2} P_{tt} \tag{1}$$

describing the propagation of a pressure wave in an acoustic medium and introduce the standard vertical traveltime variable τ defined by

$$\tau = \int_0^z \frac{dz}{v(z)} \tag{2}$$

to obtain the equation

$$v^2 P_{xx} + P_{\tau\tau} = P_{tt} \quad . \tag{3}$$

For a flat-layered medium, when P_{xx} is zero it is straightforward to verify that these vertically propagating solutions of (3) are constant along the level lines $t + \tau = constant$. For this reason you may expect that, introducing $t + \tau$ as a new variable, you may ignore higher derivatives along these level lines when dips are small. This is indeed the basis of 15° migration where Claerbout (1976) employs retarded time $t' = t + \tau$ to obtain the 15° migration equation

$$v^2 P_{xx} = -2 P_{t'\tau} \tag{4}$$

used to generate the differencing grids outlined in the introduction. The initial, unmigrated data is at $\tau = 0$ and the migrated image is at $t = 0$ or $t' = \tau$. For modeling, one may choose the alternate substitution of *retarded traveltime* $\tau' = \tau + t$ to get the classical 15° modeling equation

$$v^2 P_{xx} = +2 P_{t\tau'} \tag{5}$$

with the initial data $t = 0$ and the modeled data $\tau = 0$ or $\tau' = t$, again on the diagonal. Now,

however, the differencing star must be applied "upside down" to get a stable direction of recursion. One way to see this is to reverse the sign of τ' in equation (5) to bring it into the form of equation (4). This means turning the migrated section and imaging condition upside-down, reminiscent of Loewenthal and Mufti's 1983 reverse-time migration scheme. This formulation does not lend itself to parallelism but instead is computed recursively, as with ordinary 15° migration.

15° Finite difference modeling grid

$$
\begin{array}{cccccc}
 & A_4 & B_3 & C_2 & D_1 & E_0 \\
 & A_3 & B_2 & C_1 & D_0 & \\
 & A_2 & B_1 & C_0 & & \\
 & A_1 & B_0 & & & \\
\uparrow & A_0 & & & & \\
z & & & & & t \rightarrow
\end{array}
$$

Differencing star

$$
\begin{array}{cc}
Y_{i-1} & Z_{i-1} \\
X_i & Y_i
\end{array}
$$

FIG. 2. Alternative finite difference modeling grid. Here the depth section appears upside down on the left edge of the grid and the time section on the upside-down diagonal. The differencing star is applied upside down too.

The individual differencing star may take any one of a number of forms. The explicit 15° star may be expressed by

$$[\, Z_{i-1} + X_i \,] \;=\; (1 + 2\,\alpha\,\delta_{xx})\,[\, Y_i + Y_{i-1} \,] \tag{6}$$

where δ_{xx} is the second differencing operator and α is given by the formula

$$\alpha \;=\; \frac{v^2 \Delta t\, \Delta \tau}{8\, \Delta x^2} \quad . \tag{7}$$

The implicit 15° star is expressed by

$$(1 + (\beta - \alpha)\,\delta_{xx})\,[\, Z_{i-1} + X_i \,] \;=\; (1 + (\beta + \alpha)\,\delta_{xx})\,[\, Y_i + Y_{i-1} \,] \tag{8}$$

with the constant parameter β typically chosen to be around one sixth. The general two time level differencing star appears in the form

$$\Phi\,[\, Z_{i-1} + X_i \,] \;=\; \Psi\,[\, Y_i + Y_{i-1} \,] \tag{9}$$

where Φ and Ψ are appropriate linear operators designed to make the transfer function of (9) better match the ideal single square root response

$$k_\tau = \sqrt{\omega^2 - v^2 k_x^2} - \omega \quad . \tag{10}$$

Higher time derivatives

Higher order $x-t$ migration algorithms, e.g., 45°, 60°, etc. schemes, employ a differencing star with more time levels, e.g. a skyscraper rather than a box. This does not affect the parallelism of time-reverse ordering. It simply increases the computational cost of applying the differencing star.

Extrapolation step size

The astute reader will have noticed that all the above figures are portrayed with $\Delta\tau = \Delta t$. For reasons of efficiency and, occasionally, accuracy, $\Delta\tau$ is commonly taken to be some multiple of Δt. This reduces both the number of computations and the degree of parallelism by the same factor. For $\Delta\tau = 2\,\Delta t$ we have the situation depicted in Figure 3.

15° Finite difference migration grid, $\Delta\tau = 2\,\Delta t$

$$\begin{array}{llll} t & A_0 & & \\ \downarrow & B_0 & & \\ & C_0 & A_2 & \\ & D_0 & B_2 & \\ & E_0 & C_2 & A_4 \\ & F_0 & D_2 & B_4 \\ & & & \tau\rightarrow \end{array}$$

Differencing star

$$\begin{array}{ll} Y_{i-1} & X_i \\ Z_{i-1} & Y_i \end{array}$$

FIG. 3. Diagram of 15° finite difference migration. Each box represents a vector of spatial data values, with the input time section in the first column. Extrapolation step size is $2\,\Delta t$, giving rise to the "jagged" migrated image A_0, B_0, A_2, B_2, A_4, B_4. The differencing star is used to compute X_i.

Parallelism is retained by sweeping towards the diagonal in blocks of $\Delta\tau/\Delta t$ vertically piled vectors, as illustrated in Figure 4. Within each vertical block, however, computations must be performed in the specific, recursive order that proceeds from bottom to top.

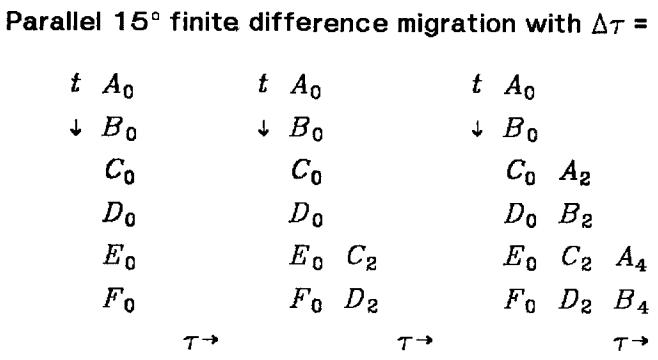

FIG. 4. Parallel 15° finite difference migration when extrapolation step size is $2\,\Delta t$. The differencing star is applied twice here to compute vertically adjacent pairs of downward continued time slices.

Francis Muir (private communication) makes the interesting point that, because parallelism is reduced in proportion to the number of computations saved by using a larger extrapolation step size, the actual time required to perform a parallel migration will be nearly independent of step size. Therefore we do not need to choose large extrapolation step sizes purely to save computational time and can freely employ the rather small extrapolation steps that are usually needed to realize the improved accuracy that high-dip algorithms offer.

A few small processors

Economic constraint may limit the number of identical differencing stars one may cram into the parallel migration device. In this event auxiliary storage will be needed to replace the lacking parallel components and to hold temporary results. A reasonable way to minimize (expensive) accesses to auxiliary storage is to mimic conventional 15° migration and do parallel sweeps as diagramed in Figure 5.

Alternatively, one may sweep horizontally instead of vertically (t-outer) or simply flip-flop processing elements along each subdiagonal, following the basic reverse time algorithm as much as possible.

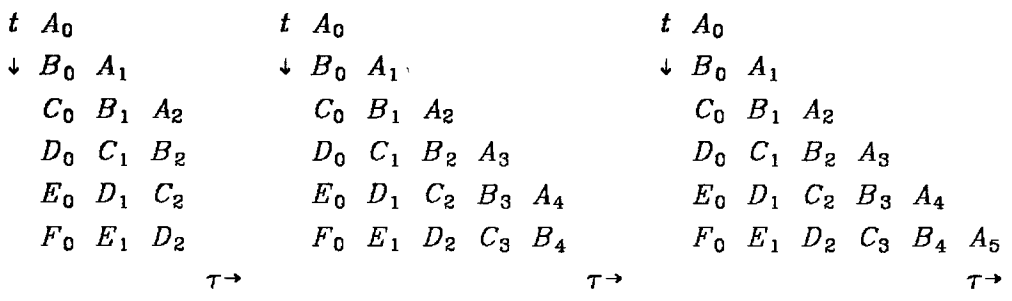

FIG. 5. Parallel 15° finite difference migration when only a limited number of processing elements are available. Here only two units are used and the migrated section is generated in tiers.

Large processors (overlap)

While it is not essential to provide each parallel unit with enough internal memory to hold the input and output vectors (and any temporary intermediate vectors that might arise), it can be desirable to do so. When the vectors are significantly shorter than the maximum length for which internal memory is designed, it is feasible to process multiple vectors inside the differencing unit. For example, one might identify each unit with a conventional pipelined array processor where it is advantageous to process multiple vectors to minimize I/O to the host computer or other processors. Here the processing element, as illustrated in Figure 6, can fill in a rectangular array computed from the sides or bottom before requiring new input.

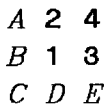

FIG. 6. Multiple input differencing star generated by repeated application of the basic differencing star. This overlapped form is useful for minimizing I/O when employing conventional large memory pipelined processors for parallel $x-t$ migration.

This allows us to sweep towards the diagonal in rectangular chunks, just as if they were simple differencing stars, only larger. Of course, edge effects will have to be accounted for.

Further parallelism

We have now seen how migration decouples into independent constant τ vector calculations in constant t (i.e., $t'-\tau$) computational planes. Explicit algorithms, such as that of equation (6), increase parallelism even further by decoupling the individual x components of each vector. This further decreases the time needed for complete migration by a factor proportional to the number of traces, say another three orders of magnitude. In rough figures, if a migration of a 1000 by 1000 point section normally takes a half an hour on a conventional mainframe, then the reverse-time organization reduces this by about a factor of $1/2\times1000 = 500$ or to just under three seconds and an explicit migration gains an additional factor of 1000, taking the time down to 3 milliseconds. Even attaining only one percent of this theoretical gain brings migration into the fold of *interactive* tools available to the practicing geophysicist.

Expense

I don't have much knowledge of the cost of systolic devices such as the ones I am proposing. I am reasonably confident, however, that since an explicit algorithm requires only a few multiplies and adds to produce its output, suitable circuitry could be produced for perhaps $250 an element. If we add this figure and the 10,000 together (giving us the 1% figure used above) to create the device, our estimated cost is around $2,500,000.

Is a separate device required?

Again I'm fairly ignorant on the state of the art in parallel machines. In the last year I've heard a couple of talks on campus by people who have designed and prototyped partially- or wholly- parallel computers. One speaker estimated that a manageable upper limit on the number of fully asynchronous, interconnectable, general-purpose processors would be around a hundred. On the other hand, systolic architectures used to implement large matrix multiplications, wherein each processor talks to only a few neighbors and output an advancing plane (or line) representing the "present" that divides the precomputed "past" from the yet to be computed "future", may be built containing thousands of independent computing elements. I think it reasonable to predict that there will appear on the market within the next few years computers capable of implementing the migration algorithms I have described with a high degree of parallelism.

Conclusions

I have shown how to introduce a high degree of parallelism into conventional time domain migration algorithms. This parallelism is even further enhanced for the class explicit migration algorithms and offers the attractive prospect of interactive migration to the geophysicist. Finally, I have speculated on how much such a device might cost and how soon developments in the field of parallel computers will produce a general purpose device capable of exploiting these parallel algorithms.

REFERENCES

Claerbout, J.F., 1976, Fundamentals of geophysical data processing with applications to petroleum prospecting: McGraw Hill, New York.

Loewenthal, D. and Mufti, I.R., 1983, Reversed time migration in spatial frequency domain: Geophysics, v. 48, p. 627-635.