147

PARALLEL ALGORITHMS FOR AUTOMATIC ESTIMATION OF SEISMIC VELOCITIES

PAUL L. STOFFA AND MRINAL K. SEN

(Principal Investigators)

AND

VIKRAMADITYA SEN

The University of Texas Institute for Geophysics
8701 North MoPac Expressway
Austin, Texas 78759-8397

UTIG TECHNICAL REPORT NO. 147

FINAL REPORT

SUBMITTED TO CRAY RESEARCH, INC., APRIL 1996

INTRODUCTION

Estimation of seismic velocities in the subsurface is a fundamental problem that geophysicists seek to solve from the analysis of seismic reflection data. The crucial relationship between accurate estimation of subsurface velocities and improved migrated seismic sections is often used to determine the validity of estimated velocity models. In other words, a properly migrated seismic section lends greater credence to the velocities used to generate it. Thus, a 'migration misfit criterion' (Jervis et al., 1996) can be formulated to quantify the shortcomings of the migrated image and relate them to the perturbations required to improve the velocity model.

Varela (1996) proposed a scheme for automatic estimation of background velocities using nonlinear optimization for reflection tomography criteria (RTC) and migration misfit criteria (MMC). The scheme has performed well for both real and synthetic 2-D data showing the feasibility of its application for the current generation of computers. In this report, we present a parallelized approach to velocity estimation based on a modification of the schemes proposed by Varela. The significant reduction in computation time for the parallelized schemes not only makes the process of velocity estimation more economical but also provides a direction for further research on applying these schemes to 3-D data.

The proposed velocity estimation scheme (Varela, 1996) performs many iterations of migration which is a computationally intensive process and the migration misfit error from each iteration is taken into account to modify the velocity model. A population of velocity models can be evaluated in this way and a nonlinear optimization scheme can be used to minimize the misfit error and lead the 'population' of models to converge towards an optimum model that resembles the true model. In a modification of Varela's approach, several of these models can be evaluated simultaneously at each 'temperature' which is a parameter for guiding the optimization.

This straightforward parallelism is identified by simultaneously evaluating several models, thus generating a coarsely parallel scheme where each processing element migrates the seismic data using a different velocity model. The migration misfit errors are later compared against each other and the velocity function optimized to reduce the misfit. In a parallel virtual machine configured under PVM, the master node reads the seismic data and all necessary computation parameters and later transmits them to the slave nodes. Each node in the virtual machine evaluates a certain number of models for MMC or RTC and returns the error values to the master node. Thus, the computationally intensive task of model evaluation is shared by the different nodes and the gain in terms of reduced computation time more than compensates for the time spent in communicating between the nodes. In this approach, the entire misfit calculation for all the data is computed in each mode for a unique trial velocity model. Communication between nodes occurs at the very start of the program (to distribute the data one time) and at each temperature change where the misfit (one real number) and the sparse velocity model (e.g., 50 real numbers) are transferred to and from the master node. This is the approach with the minimum possible communication. However, while it may be very advantageous for a cluster of networked workstations, alternatives that make efficient use of the high-speed internode communication of the Cray T3E system are also possible.

An alternate approach is based on micro-tasking where the emphasis is changed to speeding up the computationally intensive process of migration for every model evaluation. In this case, large tables of seismic traveltimes are shared and swapped among the different processing elements in order to reduce the computation time and memory requirements. This approach aims at reducing redundancy in computations and maximizing the repeated use of all intermediate computations and results. This approach requires significant communication between nodes.

This report describes a PVM based parallelization and modification of an existing sequential algorithm (Varela, 1996). The parallel algorithm is implemented through the coarse grained multi-tasking scheme.

FORMULATION OF THE VELOCITY ESTIMATION PROBLEM

In this section, we define the geophysical problem that is being solved. The principal aim is to estimate a model of the velocity distribution in the Earth (referred to as the 'subsurface velocity model') that best fits the acquired data. The data are recordings of waves usually at the surface of the Earth that are generated and transmitted in the Earth in response to a 'source' which is usually an explosion. The practice of using explosions to propagate waves in the Earth to determine its material properties is widespread in the oil industry and constitutes the 'seismic method.' A considerable portion of the resources spent in prospecting for hydrocarbons are dedicated to the efficient and accurate processing of surface seismic data and its meaningful interpretation. With the advent of high performance computers and developments in numerical analysis, barriers between qualitative interpretation and quantitative geophysical processing and analysis are being eliminated. The work of Varela (1996) and this report shows how it is possible to quantitatively estimate subsurface velocities from seismic data without subjective interpretation.

The velocity model must 'fit' the data in a satisfactory way. The data, i.e., the seismic recordings have two main aspects: the arrival time of a wave (also referred to as 'traveltime') and the amplitude of the recorded signal. In a kinematic analysis such as velocity estimation, one is concerned solely with the traveltimes. In other words, a velocity model is said to fit the data if the traveltimes computed using that velocity model for the proper source-receiver configurations agree with the traveltimes of the recorded events. Needless to say, the computation of the traveltimes forms the most important part of the velocity estimation problem. Computing traveltimes involves solving the Eikonal equation and is the most computation intensive processes involved in velocity estimation.

In most cases, as in this study, a scheme is devised to grid and parameterize the volume or section of the subsurface that is of interest. This step is referred to as 'model parameterization.' In this study, cubic splines have been used to build the velocity field by piece-wise interpolation of the velocity values at the spline node locations. In other words, the velocity values at the spline nodes are the 'parameters' that have to be estimated.

Once the model has been parameterized, traveltimes are computed and matched with the data. Based on the misfit, a perturbation factor is determined for the model based on some specified criteria. The perturbation is then applied to the model for updating and the steps of traveltime computation and comparison are repeated. The algorithm proceeds in an iterative fashion until a satisfactory agreement is reached between the recorded data and the 'synthetic data' generated from the 'velocity model.'

In this study, a direct comparison of observed and computed traveltimes is not carried out. Instead, the recorded data are 'migrated' into images and the images are compared. Migration is a procedure whereby the recorded seismic energy is mapped back to depth—i.e. an image of the subsurface is constructed using the wave equation, the recorded data, and the trial velocity model. If the velocity model is correct, the strong reflections will be mapped back into coherent images or 'reflectors' within the Earth, i.e. boundaries between different types of rocks. On the other hand, if the velocity model is only close to the true subsurface velocity distribution, any real reflecting boundary will get imaged slightly incorrectly for different source-receiver configurations, i.e. a misfit occurs. These images of the same points in the subsurface obtained from different source-receiver configurations are called a 'common image gather.' The 'misfit' referred to in this optimization study is actually a measure of disagreement determined from the mismatch among many migrated common image gathers.

OPPORTUNITIES FOR PARALLELISM

Typically, parts of existing algorithms can be parallelized while others have to remain sequential. Any parallel algorithm will (at best) have an execution time that is the sum of the execution times of its non-parallel portions. In addition, there are system overheads, system wait times etc. that prevent the program from reaching even the optimum speedup possible with N–processors. Further, efficiency is also lost whenever one or more processors remain idle. Therefore, loops in the program have to be restructured in a way that minimize overhead and maximize the actual computation times at the processing elements. In this study, an attempt was made to parallelize an existing sequential algorithm involving multiple iterations of a computationally intensive module. PVM was used as the basic tool for implementing the parallel algorithm and runs have been made on the J90 and the C90 and efforts are under way for runs on a T3D.

The optimization scheme, as shown in Figure 1, involves repeated evaluations of velocity models. With a modification of the acceptance and update criteria used by Varela (1996), a coarse grain parallelism can be identified. In other words, several models can be simultaneously evaluated at the different nodes of the parallel machine (or as in this case, the Parallel Virtual Machine) and their errors can be compared. Such a scheme has the advantage that it involves the transmission of large chunks of data (i.e. the velocity models and the recorded seismic data) only once at the beginning of the program. Since each of the model evaluations can be done completely independent of each other, this coarse parallel scheme is worth investigating (see Figures 2 and 3).

If the acceptance and update schemes of Varela (1996) are retained, an alternate parallelism can be implemented in the computation of traveltimes. This concept is illustrated in Figure 4. In any case, traveltimes at all the grid points have to be computed for a shot placed at each surface grid point. In the original sequential scheme, an attempt was made to reduce computations by actually calculating traveltimes for shots at only a few surface grid points and interpolating the tables for shots at other surface locations. Thus for nx surface points, $9 + integer((nx-11)/3)$ actual traveltime tables were computed and the rest were created through interpolation. However, for a parallelized scheme, each node has to compute fewer tables so that all the necessary traveltime tables could actually be computed, eliminating the need for any interpolation whatsoever. A comparison of the required calls to the traveltime computation subroutine is given in Figure 5.

The difference between the required number of calls at each node and the number of calls in the sequential algorithm increases as the grid is enlarged or the number of nodes is increased. However, even with a modest 4-node configuration, the differences warrant a parallelization of traveltime computations.

Yet another more sophisticated implementation can be made through 'microtasking.' As an extension of the parallelization of the traveltime computation, as mentioned above, we can parallelize both the traveltime computations and the migrations within each iteration or model evaluation. In this approach, subsets of the data are distributed to the various nodes and they are migrated in situ and later summed together to generate the final migrated images which are used for evaluation of the misfit. The computation of traveltimes is also shared by the nodes but now requires the passing of the traveltimes among all the nodes, all of which participate in the migration. It is evident that this approach needs a platform with excellent message passing capability for implementation in a feasible manner. A schematic diagram in Figure 6 shows one way in which this scheme can be visualized.

Our current implementation was only the first approach. In the original sequential algorithm, there are three main loops: over depth relaxation; over temperature; and over trials per temperature. In the innermost loop, the computation intensive migration and traveltime computations are performed for each trial per temperature. With a modification in Varela's method of update, a

coarse grained parallelization has been implemented at this level, i.e. several trials are made at each temperature with each node carrying out the evaluation of one model that is a candidate for being the best-fit model. Inside the loop over temperatures, comparison of errors is performed and acceptance criteria are applied for the models. But unlike the original sequential scheme, where each update was based on the immediate predecessor, we evaluate several models at each temperature and compare them with one another and then carry out an update. This lends a genetic algorithm (GA) like characteristic to the parallel algorithm as opposed to the VFSA (very fast simulated annealing) approach adopted by Varela for the sequential approach. In fact, this approach is closely related to a scheme called Simulated Annealing without rejected moves (Sen and Stoffa, 1995) in which VFSA model generation and cooling schedules were used. The process of error comparison, acceptance and update is fast and can be carried out at one processing element. Hence this stage is implemented as a sequential module. Similarly, the outermost loop over depth relaxations is also retained as a sequential module since the level of relaxations is usually less than three and the gains made through parallelization are not significant.

Each of the 'slave' nodes has to exchange minimal information with the 'master' node only several times, but it is extremely important that the message passing be synchronized. Such a synchronizing can be implemented by calling 'pvmfbarrier' from the PVM library and putting up barriers that must be cleared by all the nodes as a group.

RESULTS OF PARALLELIZATION

The parallelized version of the algorithm has been implemented under PVM on several different platforms. Tests have been performed on the Cray J90/16 at the High Performance Computing Facility of the University of Texas and on the Cray C90/16 belonging to the CCN network. The clock rates for these machines are 10 ns and 4 ns respectively. Efforts are under way to port and test the codes on a T3D on the CCN network.

The 2-PE run on the J90 is summarized in Tables 1-2. This run was achieved with 6.2% of the 16-CPU machine. The memory and time requirements of the original sequential algorithm being substantial, the problem was scaled down to enable quick tests to be performed and to evaluate the parallelization. Therefore, the present tests have been performed with only two offset groups from the nine used by Varela (1996). The user CPU seconds accrued at the two nodes were 7253.7971 and 5435.3147 respectively. These statistics represent a moderately good performance with a certain degree of load imbalance that can be expected for shared memory systems.

The 2-PE run on the C90 is summarized in Tables 3-4 and the 4-PE run on the C90 is summarized in Tables 5-6. In each of these cases, the number of models evaluated at each temperature equals the number of PE's. For the 2-PE run, CPU times of 3606.0529 and 2694.0714 seconds were acquired at the two nodes. For the 4-PE run, the master node required 3656.8782 CPU seconds while the slave nodes used 2686.4517 and 2689.4576 CPU seconds. The performance statistics for the 2-PE and 4-PE runs are almost similar because the load per node is the same in each case. As explained earlier, when we have more nodes in the parallel virtual machine, we evaluate more models and perform a more exhaustive search of the model space. It is prudent to adopt this scheme because the 'independence' of computations exists at the level of model evaluations per temperature (for our GA like modified algorithm) and no higher than that in the basic 3-loop sequential structure. However, as a consequence more than twice as many logical I/O requests (i.e. 17631) are made for the 4-PE run on the C90 as compared to the 2-PE runs on the C90 and the J90 (6714 and 6754 I/O requests respectively).

The issue of determining the maximum number of PE's that will justify building a larger virtual machine has to be addressed separately. First, the basic problem that we are aiming to solve is an optimization problem, i.e. a search for the 'best model.' That brings us to the trade-off between computation time and the extensiveness of the search for the best model. If more models are

evaluated at each temperature, the search is more extensive but the dynamics of the update and acceptance rules will determine a limit which will be sufficient to reach the answer and beyond which, increasing the number of evaluations at each temperature will not necessarily improve the results by any significant amount.

In Akbar et. al. (1996) we described the plane wave Kirchhoff migration method that we developed. In this approach data are first transformed to the (τ–p) domain. Constant plane wave sections are then migrated independently using the same velocity function. Images resulting from plane wave sections can be combined to form the final image just as constant offset sections are composited. Before stacking the plane wave sections, they can also be compared to obtain a velocity model that is consistent with the data. Like the conventional Kirchhoff algorithm, the new algorithm also lends itself nicely to parallelization using either the coarse grained approach described above or micro-tasking. A parallel code has been developed using PVM. Some of the results were presented at a recent PVM conference (Akbar et al., 1994; Sen et al., 1994).

Future work should investigate the microtasking approach, where the traveltime computation and migration are distributed among the available nodes. This would have the following advantages: 1) less memory per node would be required; 2) the number of nodes in use would be arbitrary and effect only the final elapsed time; and 3) the optimization parameters in terms of the number of overall evaluations per temperature and the number of temperatures would be independent of the number of nodes available. The principal disadvantage of this approach is the significant increase in communication between nodes that would be required.

Acknowledgments: This work was supported by Cray Research, Inc. Computing resources for this work were provided by The University of Texas System High Performance Computing Facility.

REFERENCES

Akbar, F. E., M. K. Sen, and P. L. Stoffa, 1996, Prestack plane wave Kirchhoff depth migration, Geophysics, in press.

Akbar, F.E., M.K. Sen, and P.L. Stoffa, 1994, Rapid 2-D Plane Wave Kirchhoff Migration using PVM, 1994, PVM Users' Group Meeting, Oak Ridge, TN.

Jervis, M., Sen, M.K., and Stoffa, P.L., 1996, Prestack migration velocity estimation using nonlinear methods, *Geophysics*, 61, 1, 138–150.

Sen, M.K., and Stoffa, P.L., 1995, *Global Optimization Methods in Geophysical Inversion*, Elsevier Science Publishing Company, The Netherlands.

Sen, V., M.. Sen and P. L.. Stoffa, 1994, Seismic Processing Using PVM: Prestack 3D Kirchhoff Migration and Modeling, 1994, PVM Users' GroupMeeting, Oak Ridge, TN.

Varela, Carlos Lopo, 1996, *Automatic Background Velocity Estimation in 2D Laterally Varying Media*, PhD Dissertation, The University of Texas at Austin.

```
 UNICOS 8.0.4.2 CRAY J916/16-4096 sn9086 (pluto.cc.utexas.edu)

 J90 CPU charges per SBU hour are:
   Standard=$18 Express=$36 Interactive=$36 Premium=$18 Premium interactive $18

 REAL DOLLAR charges for disk space are:
   $0.02 per day for the first 1 Mbyte
   $0.002 per Mbyte per day for files exempt from migration using .keep
   $0.0001 per Mbyte per day for the first 15Gb of online and migrated files
  $0.00003 per Mbyte per day for storage in excess of 15Gb
 Type "charges -d" to display your current daily charges.

 The blocktime queue is now ready for use. The MINIMUM CHARGE is $144 per
 wall-clock hour. Please read /common/doc/J90/blocktime VERY CAREFULLY
 before submitting jobs to the blocktime queue.


 For more information read /common/doc/J90/rates
                           /common/doc/J90/batch.info
                           /common/doc/J90/j90_info
                           /common/doc/J90/blocktime

 Please send problem reports to remark@hpcf.cc.utexas.edu.
 news: PrgEnv.news CF9020.news CrayLibs20.news CC20.news CrayTools20.news
  emulator10.news CrayLibs12.news
 Mon Feb 26 23:18:06 + PVM_ROOT=/hpcf/u0/go/it/goit727/pvm3
 Mon Feb 26 23:18:06 + export PVM_ROOT
 Mon Feb 26 23:18:06 + PVM_ARCH=CRAY
 Mon Feb 26 23:18:06 + export PVM_ARCH
 Mon Feb 26 23:18:06 + PVM_HALT=halt
 Mon Feb 26 23:18:06 + export PVM_HALT
 Mon Feb 26 23:18:06 + PVM_QUIT=quit
 Mon Feb 26 23:18:06 + export PVM_QUIT
 Mon Feb 26 23:18:06 + cd /hpcf/u0/go/it/goit727/pvm3/bin/CRAY
 Mon Feb 26 23:18:06 + echo halt
 Mon Feb 26 23:18:06 + 1> pvm_halt
 Mon Feb 26 23:18:06 + echo quit
 Mon Feb 26 23:18:06 + 1> pvm_quit
 Mon Feb 26 23:18:06 + ja
 Mon Feb 26 23:18:06 + pvm
 Mon Feb 26 23:18:06 + 0< pvm_quit
 pvm>
 pvmd still running.
 Mon Feb 26 23:18:08 + /hpcf/u0/go/it/goit727/pvm3/bin/CRAY/migparmod
 Mon Feb 26 23:18:08 + 0< vfsarlxco2.in
   nodes 2

 OPTIMIZATION INPUT PARAMETERS

 initial temperature and no of trials ?
 1.,   3
  no.of moves/trial, decay rate
 nmov = 2
 2,   0.98
 max. no. of iterations without changing
 5
 resume (y=1), # of iterations and
 initial error of previous run
  2,   0,   0.

  PARAMETERIZATION INPUT PARAMETERS

  Enter # of x splines, # of z splines
  5,   10
  what kind of parameterization?
  cubic splines (type 1) or ramp model (otherwise)
  1

  COST FUNCTION

  what kind of cost function?
  M.M.C. (type 1) or R.T.C. (otherwise)
  1

  RELAXATION PARAMETERS

  Enter # of relaxation steps in depth and offset
  2,   1

  Number of Offsets
  noffset_tot=2

  Input Constant Offset Sections File Name

  Acquisition Geometry Parameters
  nx=301dx=5.E-2
  nt=501 dt=4.E-3
  nz_tot=201 dz=1.E-2
   zs=-1.51 t0=2. fc=22. iapert=20
  slo0=0.6622516556291

  Mute parameters defined for the first x location
  and changed according to the sea bottom indices
  t0s=0. slope=0. lw=201

  Sea Bottom Depth Index File Name
  DRAWING INITIAL MODEL
  harmonic error=9.0825728415084E-2
  harmonic error=9.1773784030593E-2
  harmonic error=9.1093717506865E-2
  harmonic error=9.0591588935602E-2
  harmonic error=9.2303734328617E-2
  harmonic error=9.1907158381481E-2
  harmonic error=9.2632654135933E-2
  harmonic error=9.1472788062141E-2
  Finished Successfully at NODE 0
  STOP executed at line 1320 in Fortran routine '$MAIN'
  CP: 7253.878s,  Wallclock: 7298.825s,  6.2% of 16-CPU Machine
  HWM mem: 41387879, HWM stack: 3336205, Stack overflows: 0
 Tue Feb 27 01:19:48 + pvm
 Tue Feb 27 01:19:48 + 0< pvm_halt
 pvmd already running.
 pvm> libpvm [t40006]: mxfer() EOF on pvmd sock
 Tue Feb 27 01:19:48 + ja -csftlh
```

Table 1

Job Accounting - Command Report

Command Name	Started At	Elapsed Seconds	User CPU Seconds	Sys CPU Seconds	I/O Wait Sec Lck	I/O Wait Sec Unlck	CPU MEM Avg Mwds	I/O WMem Avg Mwds	Kwords Xferred	Log I/O Request	Memory HiWater	Ex St	Ni	Fl	SBU's
ja	23:18:06	0.2632	0.0048	0.0170	0.1320	0.1102	0.0852	0.0976	0.00	0	200	0	39		0.00
pvm	23:18:07	0.0039	0.0000	0.0039	0.0000	0.0000	0.0618	0.0000	0.00	0	160	0	39	F	0.00
pvmd3	23:18:08	0.0080	0.0012	0.0054	0.0000	0.0000	0.1164	0.0000	0.01	2	288	0	39	F	0.00
pvm	23:18:06	1.4256	0.0128	0.1470	0.1076	0.0734	0.0736	0.0781	0.08	24	168	0	39		0.00
migparmo	23:18:10	7298.2567	5435.3147	31.2292	0.0000	0.0458	39.4655	0.0000	3242.75	1800	80832	0	39		0.00
migparmo	23:18:08	7300.2700	7253.7971	10.8374	0.0000	0.2238	39.4703	0.0000	3821.75	1058	80848	0	39		0.00
pvmd3	23:18:07	7302.1155	0.3618	3.4456	0.1200	0.2772	0.4414	0.1250	6499.00	3768	1536	0	39		0.00
pvmgs	23:18:10	7298.9278	0.0460	0.0972	0.1059	0.0156	0.1093	0.1035	0.28	81	296	0	39		0.00
pvm	01:19:48	0.5390	0.0102	0.1268	0.0625	0.0698	0.0725	0.0780	0.06	21	168	0	39		0.00

Job Accounting - Command Flow Report

```
parent    ( CPU time) -> child    ( CPU time) ...
================================================
ja        (   0.0219)
pvm       (   0.1598) -> pvm      (   0.0040)
pvmd3     (   3.8073) -> pvmd3    (   0.0066)
                      -> migparmo (5466.5439)
migparmo (7264.6345)
pvmgs     (   0.1432)
pvm       (   0.1370)
```

Job Accounting - Summary Report

```
Job Accounting File Name          : /nqstmp/nqs.+++++11UO/.jacct55819
Operating System                  : unicos pluto 8.0.4 8.0.2 CRAY J90
User Name (ID)                    : goit727 (5037)
Group Name (ID)                   : goit (1313)
Account Name (ID)                 : A-05037 (5037)
Job Name (ID)                     : MIGPARMOD (55819)
Report Starts                     : 02/26/96 23:18:06
Report Ends                       : 02/27/96 01:19:49
Elapsed Time                      :         7303       Seconds
User CPU Time                     :        12689.5486  Seconds
System CPU Time                   :           45.9095  Seconds
I/O Wait Time (Locked)            :            0.5279  Seconds
I/O Wait Time (Unlocked)          :            0.8158  Seconds
CPU Time Memory Integral          :       502478.5208  Mword-seconds
SDS Time Memory Integral          :            0.0000  Mword-seconds
I/O Wait Time Memory Integral     :            0.0521  Mword-seconds
Data Transferred                  :           13.2460  MWords
Maximum memory used               :           39.4766  MWords
Logical I/O Requests              :         6754
Physical I/O Requests             :          366
Number of Commands                :            9
Billing Units                     :            0.0000
```

Table 2

```
      Cray Research, Inc. CRAY C90 UNICOS Production System - RAIN

                          UNICOS 9.0.1.2
                          Source Release

    ************************************************************
       If you experience any problems with RAIN please contact
       Information Services at 612-683-7000 and press 4.
    ************************************************************

       The plan to reconfigure the T3D (CALM) to a 64 PE system
       on Sunday 2/4/96 has been postponed. The announcement for
       the reconfiguration will come at a later date.


You have mail.
news: PrgEnv.news CF9020.news prod.news CF77_M62.news MPT10.news CF90_M01.news
 CC20.news CrayTools20.news CrayLibs20.news

>> Cpu quota assigned - 18000.0 seconds
>> You have used 139.7 seconds for this month.
>> If you have any questions about this quota limit, contact your sponsor.

Wed Mar  6 19:03:04 + PATH=/bin:/usr/bin:/usr/ucb:/usr/lbin:./
Wed Mar  6 19:03:04 + export PATH
Wed Mar  6 19:03:04 + /usr/spool/nqs/scripts/+++LO++++0w+++
Wed Mar  6 19:03:04 + PVM_ROOT=/rain/u9/n4833/pvm3
Wed Mar  6 19:03:04 + export PVM_ROOT
Wed Mar  6 19:03:04 + PVM_ARCH=CRAY
Wed Mar  6 19:03:04 + export PVM_ARCH
Wed Mar  6 19:03:04 + PVM_HALT=halt
Wed Mar  6 19:03:04 + export PVM_HALT
Wed Mar  6 19:03:04 + PVM_QUIT=quit
Wed Mar  6 19:03:04 + export PVM_QUIT
Wed Mar  6 19:03:04 + cd /rain/u9/n4833/pvm3/bin/CRAY
Wed Mar  6 19:03:04 + echo halt
Wed Mar  6 19:03:04 + 1> pvm_halt
Wed Mar  6 19:03:04 + echo quit
Wed Mar  6 19:03:04 + 1> pvm_quit
Wed Mar  6 19:03:05 + ja
Wed Mar  6 19:03:05 + pvm
Wed Mar  6 19:03:05 + 0< pvm_quit
pvm>
pvmd still running.
Wed Mar  6 19:03:06 + /rain/u9/n4833/pvm3/bin/CRAY/migparmod
Wed Mar  6 19:03:06 + 0< vfsarlxco2.in
  nodes 2

 OPTIMIZATION INPUT PARAMETERS

 initial temperature and no of trials ?
 1.,  3
  no.of moves/trial, decay rate
 nmov = 2
 2,  0.9800000000000004
 max. no. of iterations without changing
 5
  resume (y=1), # of iterations and
  initial error of previous run
  2,  0,  0.

  PARAMETERIZATION INPUT PARAMETERS

  Enter # of x splines, # of z splines
  5,  10
  what kind of parameterization?
  cubic splines (type 1) or ramp model (otherwise)
  1

  COST FUNCTION

  what kind of cost function?
  M.M.C. (type 1) or R.T.C. (otherwise)
  1

  RELAXATION PARAMETERS

  Enter # of relaxation steps in depth and offset
  2,  1

  Number of Offsets
  noffset_tot=2

  Input Constant Offset Sections File Name

  Acquisition Geometry Parameters
  nx=301dx=5.000000000000004E-2
  nt=501 dt=4.000000000000003E-3
  nz_tot=201 dz=1.E-2
   zs=-1.509999999999998 t0=2. fc=22. iapert=20
  slo0=0.6622516556291416

  Mute parameters defined for the first x location
  and changed according to the sea bottom indices
  t0s=0. slope=0. lw=201

  Sea Bottom Depth Index File Name
  DRAWING INITIAL MODEL
  harmonic error=9.082572841508351E-2
  harmonic error=9.177378403059322E-2
  harmonic error=9.109371750686534E-2
  harmonic error=9.059158893560237E-2
  harmonic error=9.230373432861682E-2
  harmonic error=9.190715838148122E-2
  harmonic error=9.263265413593302E-2
  harmonic error=9.147278806214131E-2
  Finished Successfully at NODE 0
  STOP executed at line 1300 in Fortran routine '$MAIN'
  CP: 3606.354s,  Wallclock: 5705.378s,  4.0% of 16-CPU Machine
  HWM mem: 41398339, HWM stack: 3336197, Stack overflows: 0
Wed Mar  6 20:38:13 + pvm
Wed Mar  6 20:38:13 + 0< pvm_halt
pvmd already running.
pvm> libpvm [t40006]: mxfer() EOF on pvmd sock
Wed Mar  6 20:38:13 + ja -csftlh
```

Table 3

Job Accounting - Command Report

Command Name	Started At	Elapsed Seconds	User CPU Seconds	Sys CPU Seconds	I/O Wait Sec Lck	I/O Wait Sec Unlck	CPU MEM Avg Mwds	I/O WMem Avg Mwds	Kwords Xferred	Log I/O Request	Memory HiWater	Ex St	Ni	Fl	SBU's
ja	19:03:05	0.0507	0.0016	0.0072	0.0002	0.0000	0.1765	0.2142	0.00	0	448	0	23		0.00
pvm	19:03:05	0.0033	0.0000	0.0016	0.0000	0.0000	0.0644	0.0000	0.00	0	176	0	23	F	0.00
pvmd3	19:03:06	0.0080	0.0004	0.0019	0.0000	0.0000	0.1112	0.0000	0.01	2	272	0	23	F	0.00
pvm	19:03:05	1.5367	0.0039	0.0183	0.0572	0.0455	0.0868	0.0859	0.08	24	192	0	23		0.00
migparmo	19:03:07	5705.3719	2694.0714	37.3075	0.1708	0.0733	39.4991	36.3046	3242.69	1802	80896	0	23		0.00
migparmo	19:03:06	5705.9312	3606.0529	52.1350	1.4135	4.0505	39.4920	38.0283	3821.63	1030	80896	0	23		0.00
pvmd3	19:03:05	5708.7274	0.2717	1.5509	1.1727	0.1468	1.7676	2.5647	6642.50	3754	6096	0	23		0.00
pvm	20:38:13	0.6844	0.0031	0.0166	0.1379	0.3342	0.0866	0.0898	0.06	21	192	0	23		0.00
pvmgs	19:03:07	5706.4904	0.0152	0.0300	0.0772	0.0053	0.1230	0.1172	0.28	81	272	0	23		0.00

Job Accounting - Command Flow Report

```
parent    ( CPU time) -> child     ( CPU time) ...
================================================
ja        (   0.0087)
pvm       (   0.0222) -> pvm       (   0.0016)
pvmd3     (   1.8225) -> pvmd3     (   0.0023)
                      -> migparmo (2731.3788)
migparmo (3658.1878)
pvmgs     (   0.0452)
pvm       (   0.0197)
```

Job Accounting - Summary Report

```
Job Accounting File Name          : /tmp/nqs.++++0w+LO/.jacct16755
Operating System                  : sn4025 rain 9.0.1ai prd.7 CRAY C90
User Name (ID)                    : n4833 (25339)
Group Name (ID)                   : n3024 (11704)
Account Name (ID)                 : n4833 (25339)
Job Name (ID)                     : MIGPARMOD (16755)
Report Starts                     : 03/06/96 19:03:05
Report Ends                       : 03/06/96 20:38:13
Elapsed Time                      :         5708      Seconds
User CPU Time                     :         6300.4202 Seconds
System CPU Time                   :           91.0688 Seconds
I/O Wait Time (Locked)            :            3.0296 Seconds
I/O Wait Time (Unlocked)          :            4.6557 Seconds
CPU Time Memory Integral          :       252359.4994 Mword-seconds
SDS Time Memory Integral          :            0.0000 Mword-seconds
I/O Wait Time Memory Integral     :           62.9897 Mword-seconds
Data Transferred                  :           13.3860 MWords
Maximum memory used               :           39.5000 MWords
Logical I/O Requests              :         6714
Physical I/O Requests             :          223
Number of Commands                :            9
Billing Units                     :            0.0000
```

Table 4

```
      Cray Research, Inc. CRAY C90 UNICOS Production System - RAIN

                          UNICOS 9.0.1.2
                          Source Release

    ************************************************************
       If you experience any problems with RAIN please contact
       Information Services at 612-683-7000 and press 4.
    ************************************************************

       The plan to reconfigure the T3D (CALM) to a 64 PE system
       on Sunday 2/4/96 has been postponed. The announcement for
       the reconfiguration will come at a later date.


You have mail.
news: CrayTools20.news PrgEnv.news CrayLibs20.news CF77_M62.news CF9020.news
 prod.news MPT10.news CF90_M01.news

>> Cpu quota assigned - 18000.0 seconds
>> You have used 14.6 seconds for this month.
>> If you have any questions about this quota limit, contact your sponsor.

Thu Mar 14 18:03:08 + PATH=/bin:/usr/bin:/usr/ucb:/usr/lbin:./
Thu Mar 14 18:03:08 + export PATH
Thu Mar 14 18:03:08 + /usr/spool/nqs/scripts/++0LE++++0w+++
Thu Mar 14 18:03:08 + PVM_ROOT=/rain/u9/n4833/pvm3
Thu Mar 14 18:03:08 + export PVM_ROOT
Thu Mar 14 18:03:08 + PVM_ARCH=CRAY
Thu Mar 14 18:03:08 + export PVM_ARCH
Thu Mar 14 18:03:08 + PVM_HALT=halt
Thu Mar 14 18:03:08 + export PVM_HALT
Thu Mar 14 18:03:08 + PVM_QUIT=quit
Thu Mar 14 18:03:08 + export PVM_QUIT
Thu Mar 14 18:03:08 + cd /rain/u9/n4833/pvm3/bin/CRAY
Thu Mar 14 18:03:08 + echo halt
Thu Mar 14 18:03:08 + 1> pvm_halt
Thu Mar 14 18:03:08 + echo quit
Thu Mar 14 18:03:08 + 1> pvm_quit
Thu Mar 14 18:03:08 + ja
Thu Mar 14 18:03:08 + pvm
Thu Mar 14 18:03:08 + 0< pvm_quit
pvm>
pvmd still running.
Thu Mar 14 18:03:09 + /rain/u9/n4833/pvm3/bin/CRAY/migparmod
Thu Mar 14 18:03:09 + 0< vfsarlxco2.in
  nodes 4

 OPTIMIZATION INPUT PARAMETERS

 initial temperature and no of trials ?
 1.,  3
  no.of moves/trial, decay rate
 nmov = 4
 4,  0.9800000000000004
 max. no. of iterations without changing
 5
 resume (y=1), # of iterations and
 initial error of previous run
 2,  0,  0.

 PARAMETERIZATION INPUT PARAMETERS

 Enter # of x splines, # of z splines
 5,  10
 what kind of parameterization?
 cubic splines (type 1) or ramp model (otherwise)
 1

 COST FUNCTION

 what kind of cost function?
 M.M.C. (type 1) or R.T.C. (otherwise)
 1

 RELAXATION PARAMETERS

 Enter # of relaxation steps in depth and offset
 2,  1

 Number of Offsets
 noffset_tot=2

 Input Constant Offset Sections File Name

 Acquisition Geometry Parameters
 nx=301dx=5.000000000000004E-2
 nt=501 dt=4.000000000000003E-3
 nz_tot=201 dz=1.E-2
  zs=-1.509999999999998 t0=2. fc=22. iapert=20
 slo0=0.6622516556291416

 Mute parameters defined for the first x location
 and changed according to the sea bottom indices
 t0s=0. slope=0. lw=201

 Sea Bottom Depth Index File Name
 DRAWING INITIAL MODEL
 harmonic error=9.082572841508351E-2
 harmonic error=9.14245961496789E-2
 harmonic error=8.933019464642791E-2
 harmonic error=9.186343036816424E-2
 harmonic error=9.08987642345167E-2
 harmonic error=8.900753059918997E-2
 harmonic error=8.97728053940412E-2
 harmonic error=8.828083586654855E-2
 Finished Successfully at NODE 0
 STOP executed at line 1300 in Fortran routine '$MAIN'
 CP: 3590.590s,  Wallclock: 10182.639s,  2.2% of 16-CPU Machine
 HWM mem: 41398339, HWM stack: 3336197, Stack overflows: 0
Thu Mar 14 20:52:52 + pvm
Thu Mar 14 20:52:52 + 0< pvm_halt
pvmd already running.
pvm> libpvm [t4000a]: mxfer() EOF on pvmd sock
Thu Mar 14 20:52:53 + ja -csftlh
```

Table 5

Job Accounting - Command Report

Command Name	Started At	Elapsed Seconds	User CPU Seconds	Sys CPU Seconds	I/O Wait Sec Lck	I/O Wait Sec Unlck	CPU MEM Avg Mwds	I/O WMem Avg Mwds	Kwords Xferred	Log I/O Request	Memory HiWater	Ex St	Ni	Fl	SBU's
ja	18:03:08	0.0074	0.0015	0.0058	0.0002	0.0000	0.1804	0.2097	0.00	0	448	0	23		0.00
pvm	18:03:08	0.0017	0.0000	0.0016	0.0000	0.0000	0.0627	0.0000	0.00	0	176	0	23	F	0.00
pvmd3	18:03:09	0.0036	0.0004	0.0015	0.0000	0.0000	0.1056	0.0000	0.01	2	272	0	23	F	0.00
pvm	18:03:08	1.1599	0.0039	0.0174	0.0012	0.0093	0.0872	0.0840	0.08	24	192	0	23		0.00
pvmgs	18:03:10	0.0558	0.0020	0.0121	0.0004	0.0053	0.1193	0.1044	0.06	17	272	0	23		0.00
migparmo	18:03:10	10179.8554	2686.4517	46.1920	0.1952	0.0744	39.4995	36.3030	3242.69	1794	80896	0	23		0.00
migparmo	18:03:09	10183.2109	3590.3242	66.5540	1.7420	18.8809	39.4922	36.3537	10306.00	2692	80896	0	23		0.00
migparmo	18:03:10	10182.6516	2689.4576	45.2177	0.1726	0.0269	39.4990	36.3060	3242.75	1812	80896	0	23		0.00
pvmd3	18:03:08	10184.3294	0.4482	5.1842	0.0437	0.2639	2.8980	0.1745	19616.00	11150	7424	0	23		0.00
pvm	20:52:52	0.2405	0.0031	0.0162	0.0577	0.1362	0.0867	0.0859	0.06	21	192	0	23		0.00
pvmgs	18:03:10	10182.6516	0.0220	0.0382	0.0768	0.0057	0.1233	0.1172	0.42	119	272	0	23		0.00

Job Accounting - Command Flow Report

```
parent    ( CPU time) -> child    ( CPU time) ...
================================================
ja        (   0.0074)
pvm       (   0.0213) -> pvm      (   0.0016)
pvmd3     (   5.6324) -> pvmd3    (   0.0018)
                      -> pvmgs    (   0.0141)
                      -> migparmo (2732.6437)
                      -> migparmo (2734.6753)
migparmo (3656.8782)
pvmgs     (   0.0602)
pvm       (   0.0193)
```

Job Accounting - Summary Report

```
Job Accounting File Name          : /tmp/nqs.++++0w0LE/.jacct72740
Operating System                  : sn4025 rain 9.0.1ai prd.7 CRAY C90
User Name (ID)                    : n4833 (25339)
Group Name (ID)                   : n3024 (11704)
Account Name (ID)                 : n4833 (25339)
Job Name (ID)                     : MIGPARMOD (72740)
Report Starts                     : 03/14/96 18:03:08
Report Ends                       : 03/14/96 20:52:52
Elapsed Time                      :        10184      Seconds
User CPU Time                     :      8966.7147    Seconds
System CPU Time                   :       163.2406    Seconds
I/O Wait Time (Locked)            :         2.2896    Seconds
I/O Wait Time (Unlocked)          :        19.4026    Seconds
CPU Time Memory Integral          :    360389.6705    Mword-seconds
SDS Time Memory Integral          :         0.0000    Mword-seconds
I/O Wait Time Memory Integral     :        76.7020    Mword-seconds
Data Transferred                  :        35.5548    MWords
Maximum memory used               :        39.5000    MWords

Logical I/O Requests              :       17631
Physical I/O Requests             :         287
Number of Commands                :          11
Billing Units                     :           0.0000
```

Table 6

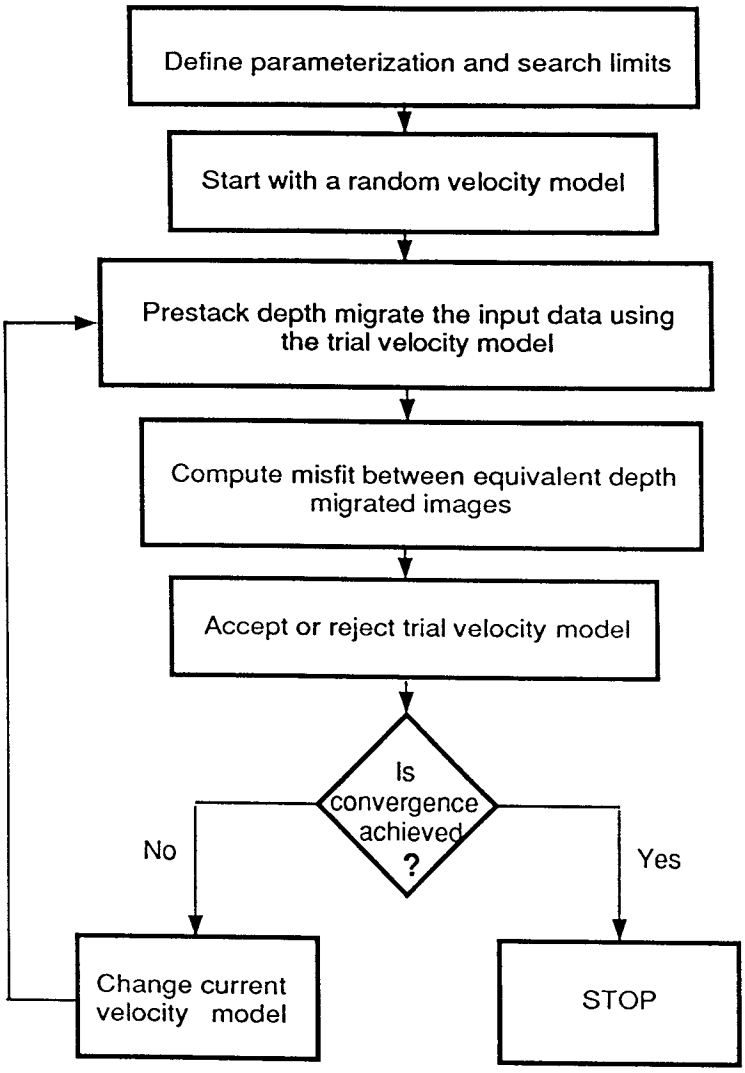
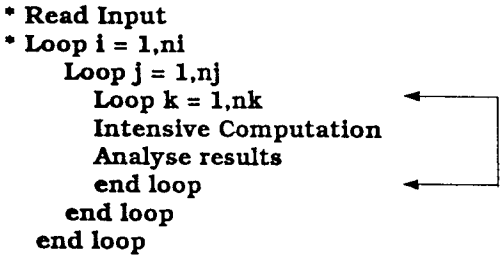


Figure 1: The automatic MMC velocity estimation scheme (Varela,1996)

Serialised Scheme



Coarse grained parallelisation

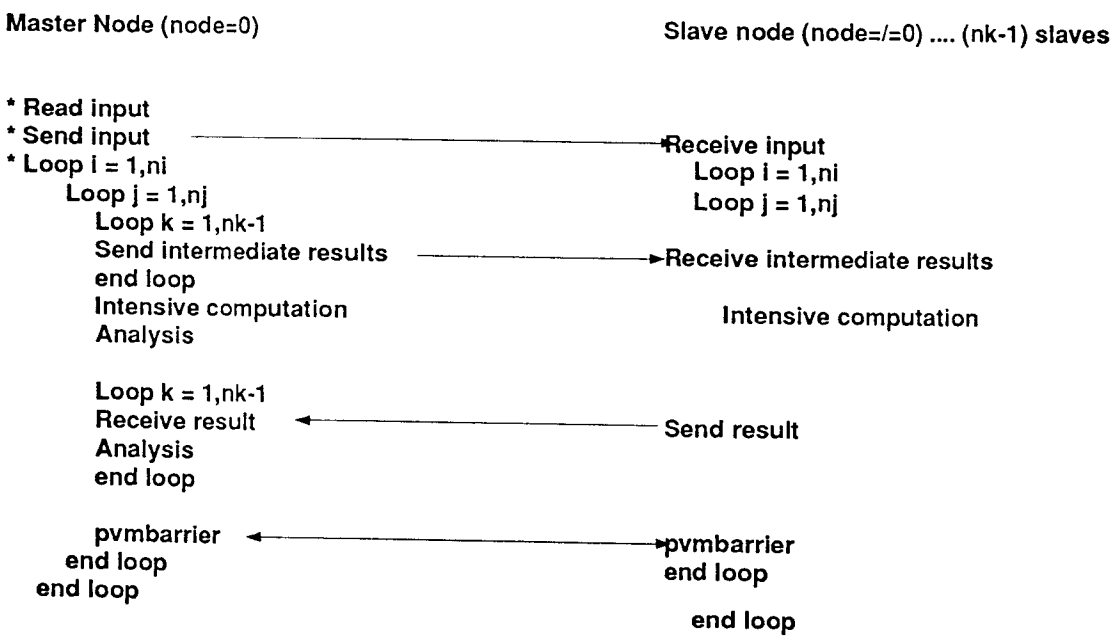


Figure 2: The modified coarse-grained scheme based on multi-tasking

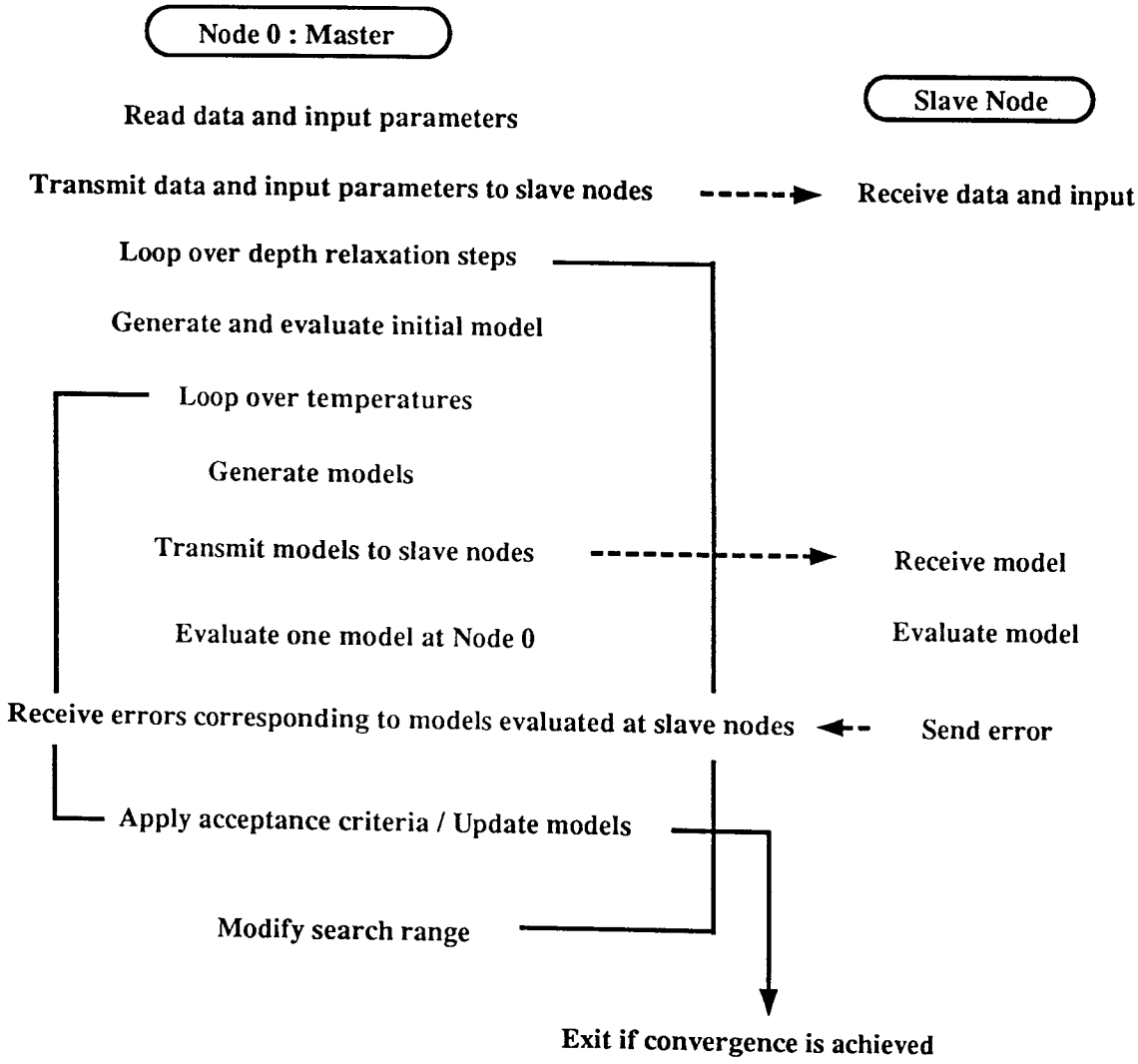


Figure 3: The modified coarse-grained scheme based on multi-tasking

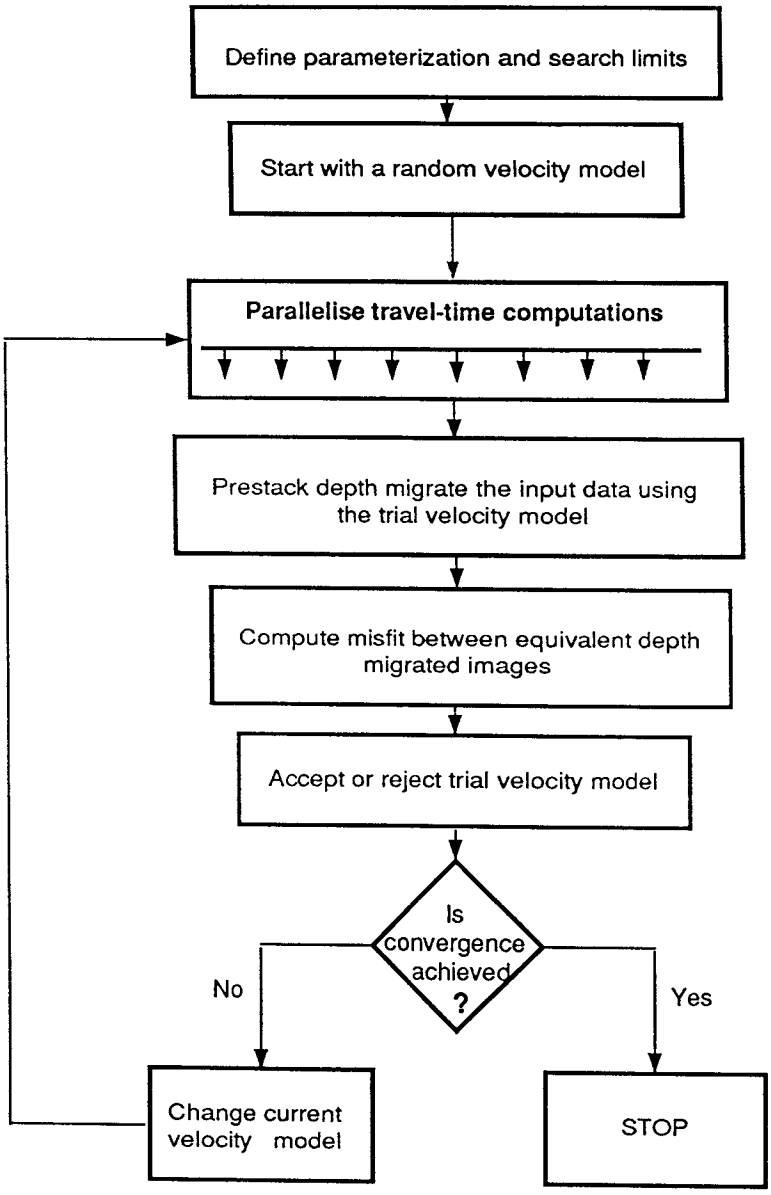


Figure 4: Parallelisation of travel-time computations

Number of calls made to the travel-time computation subroutine in Varela's scheme is compared here with the number of calls made in a most equitable work-sharing scheme for a modest 4-node parallel configuration. With the involvement of more nodes , the work load of each node will progressively decrease. If the total number of surface grid points is not a multiple of 4, one or more nodes in the parallel scheme perform an extra computation.

Number of surface grid points	Calls in Varela's scheme	Calls at the 4 nodes (0-3)
25	13	6,6,6,7
50	22	12,12,13,13
100	38	25,25,25,25
200	72	50,50,50,50

Figure 5: Parallelisation enables us to compute more travel-time tables in less time. Moreover, no interpolation is required in the parallel scheme.

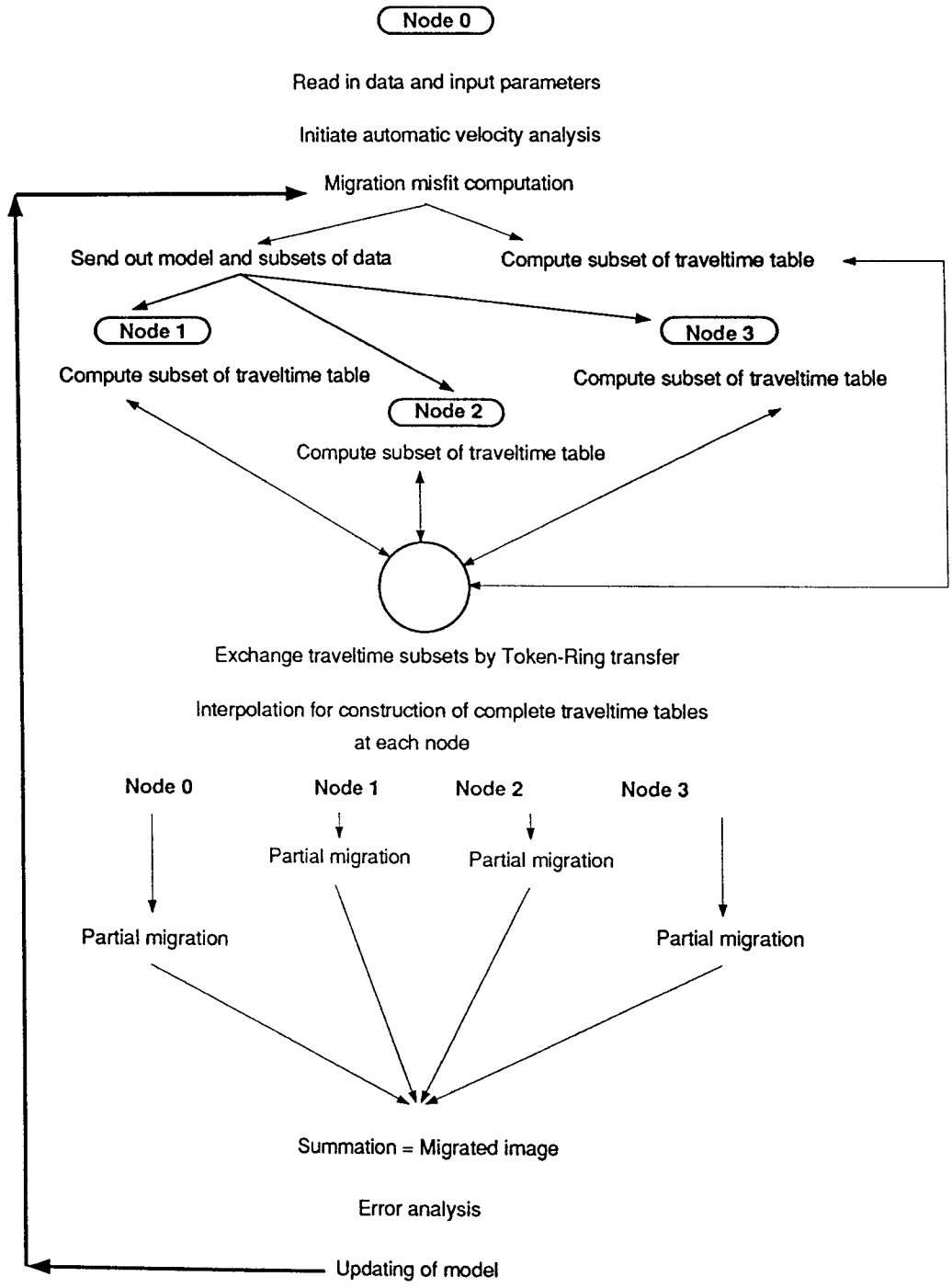


Figure 6: Visualisation of a microtasking approach