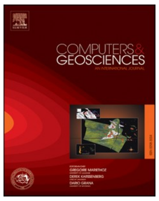
Computers & Geosciences 131 (2019) 102–111

Contents lists available at ScienceDirect

Computers and Geosciences

journal homepage: www.elsevier.com/locate/cageo



An optimized parallelized SGFD modeling scheme for 3D seismic wave propagation

Ning Wang[a], Hui Zhou[a,*,1], Hanming Chen[a], Yufeng Wang[a], Jinwei Fang[a], Pengyuan Sun[b], Jianlei Zhang[b], Yukun Tian[c]

[a] *State Key Laboratory of Petroleum Resources and Prospecting, CNPC Key Lab of Geophysical Exploration, China University of Petroleum, 102249, Changping, Beijing, China*
[b] *BGP Research and Development Center, 07275 Zhuozhou, Hebei, China*
[c] *Oil & Gas Survey, CGS, Beijing 100083, China*

A B S T R A C T

Large-scale three-dimensional (3D) seismic modeling is considered the foundation of imaging and inversion. Parallelization strategies are crucial to solving such a computationally intensive problem. In this paper, we focus on two factors, decomposition direction and decomposition dimension, that significantly affect the computational performance. The decomposition direction determines the cache hit ratio during register addressing, and the decomposition dimension influences the communication size. We thoroughly analyze these two factors by performing time-space domain staggered-grid finite-difference (SGFD) modeling with a set of decomposition strategies. Four metrics, including computation time, speedup ratio, strong scaling property, and memory usage, are introduced to evaluate the computational performance of each trial. After theoretical analysis and metrics testing, we conclude that the optimized domain decomposition strategy is: decomposing models at two dimensions, the decomposition directions are perpendicular to the fastest and the second fastest dimensions, here we refer the dimension in which data are continuously saved as the fastest dimension. Three examples further verify the feasibility and efficiency of the optimized parallel scheme. Considering that domain decomposition-based 3D seismic parallel simulation packages are seldom available in the public domain, we provide a program template for the optimized domain decomposition strategies as an open-source package.

1. Introduction

Accurate and efficient numerical techniques are required in seismic wavefield simulations (Etgen and O`Brien, 2007; O`Brien et, al., 2009; Huang and Dong, 2009a, 2009b; Chen et al., 2017; Guo and McMechan, 2017; Wang et al., 2018a; Shukla et al., 2019), reverse time migration (Wang et al., 2017, 2018b; Guo and McMechan, 2018; Q. Zhao et al., 2018; Cai et al., 2019), and full waveform inversion (Virieux and Operto, 2009; Raknes and Amtsen, 2016; Zhang et al., 2018a; Qu et al., 2019). Among the various numerical methods, the finite-difference (FD) method is commonly used due to its easy implementation and suitability for parallel computing (Dablain, 1986). Current research on FD wavefield simulations mainly focuses on three aspects: improving the simulation accuracy, solving more complex wave equations that are similar to realistic situations, and reducing the computational cost.

In general, the computation cost will increase when the simulation accuracy is improved or more complex wave equations are considered, especially for large-scale 3D models. Decomposing the computational tasks onto distributed clusters is a common approach to address such a computationally intensive problem. Many techniques in geophysics, including wave propagation modeling (Gao and Zhang, 2006; Sheen et al., 2006; Weiss and Shragge, 2013; Rubio et al., 2014), inverse problem solving (Zhang et al., 2015; Gokhberg and Fichtner, 2016; Fang et al., 2018; Chen et al., 2018a, 2018b; Li et al., 2019), migration imaging (Yang et al., 2014; Lindstrom et al., 2016; Zhang et al., 2018b; Guan and Niu, 2017, 2018), and image processing (Chen et al., 2018, 2019; X. Zhao et al., 2018), have benefited from central processing unit (CPU) clusters or graphics processing unit (GPU) clusters. Due to massive parallelism and memory hierarchy architecture, the computing speed of GPUs is faster than that of CPUs. However, the limited memory size and higher energy consumption are bottlenecks for GPUs (Said et al., 2017). Compared with GPUs, CPUs are usually free from memory limits, and CPU clusters are more prevalent. Many researchers have developed parallel algorithms based on CPU clusters. For instance, Olsen

* The corresponding author.
E-mail address: huizhou@cup.edu.cn (H. Zhou).
[1] Authorship statements: Ning Wang developed the code package, carried out the simulations, performed results analysis and wrote the draft manuscript. Hui Zhou performed the results analysis and revised the manuscript. Hanming Chen developed the code and performed results analysis. Yufeng Wang, Jinwei Fang and Yukun Tian carried out the simulations and analyzed results. Pengyuan Sun and Jianlei Zhang discussed the results and revised the manuscript.

https://doi.org/10.1016/j.cageo.2019.06.017
Received 3 March 2019; Received in revised form 27 June 2019; Accepted 28 June 2019
Available online 29 June 2019

(1995) proposed an efficient parallel method to simulate 2 min of long-period ground motion in the Los Angeles area, and Chu (2009) developed a parallel Fourier method and investigated the potential of using non-blocking all-to-all communications to overlap communication and computation. Several single instruction multiple data (SIMD)-based studies have also been conducted to further improve the parallel efficiency (e.g., Zhou and Symes, 2014). Recently, Sunway TaihuLight supercomputer-based large-scale nonlinear earthquake simulations have achieved promising results, where up to 10 million cores are employed efficiently, enabling the simulation of the 1976 Tangshan earthquake in China as an 18-Hz scenario with an 8-m resolution (Fu et al., 2017).

Domain decomposition is widely used for large-scale wavefield simulations to improve their computational efficiency. For example, Ren et al. (2014) developed a parallel domain decomposition approach for solving the Maxwell equations by the finite element method to simulate 3D large-scale electromagnetic induction in the earth. Weiss and Shragge (2013) developed a novel GPU-based parallel simulation approach for the 3D anisotropic elastic wave equation, in which domain decomposition is employed to circumvent the limited memory of an individual GPU. Etgen and O'Brien (2007) proposed an efficient out-of-core technique to improve the cache hit ratio on a single workstation. Improving the cache hit ratio is also one of the goals of this study; however, the difference is that we achieve this goal by choosing the optimal domain decomposition direction, and our scheme is designed for computations on distributed clusters. In addition, the proposed optimized domain decomposition scheme is able to reduce the communications among nodes through a reasonable decomposition dimension. This study is conducted because although decomposition strategies have dramatic effects on the computational efficiency, few studies have systematically discussed them. In this study, based on several existing classical parallel strategies, we propose an optimized domain-decomposition scheme that can make full use of the computational resources and increase the computational efficiency without additional hardware input, and we provide a program template for the optimized domain decomposition strategies as an open-source package.

The rest of this paper is organized as follows. First, we theoretically analyze the impact of the domain decomposition direction on the cache hit ratio and the effect of the domain decomposition dimension on the communication between nodes. We then investigate four performance metrics to evaluate the utility of three domain decomposition schemes and develop an optimized decomposition scheme after the theoretical analysis and performance test. Third, we describe the architecture of our code and several program optimization schemes. Finally, we test three applications on the Tianhe-1A supercomputer to demonstrate the superior properties of the proposed optimized parallel scheme and draw conclusions.

2. Methodology

2.1. Prior explanations and assumptions

In this paper, we take the following first-order 3D wave equations as an example for illustration and simulation,

$$\begin{cases} \rho \dfrac{\partial \mathbf{v}}{\partial t} - \mathbf{E}^{\mathrm{T}}\boldsymbol{\sigma} = \mathbf{0}, \\ \dfrac{\partial \boldsymbol{\sigma}}{\partial t} - \mathbf{C}\mathbf{E}\mathbf{v} = \mathbf{f}, \end{cases} \tag{1}$$

where ρ is the density, $\mathbf{v} = \begin{bmatrix} v_x & v_y & v_z \end{bmatrix}^{\mathrm{T}}$ is the particle velocity consisting of three components of $v_x,\ v_y,\ v_z$, and $\mathbf{f}$ is the source function. When

$$\begin{aligned} \boldsymbol{\sigma} &= \begin{bmatrix} \sigma_{xx} & \sigma_{yy} & \sigma_{zz} & \sigma_{yz} & \sigma_{xz} & \sigma_{xy} \end{bmatrix}^{\mathrm{T}}, \\ \mathbf{E} &= \begin{bmatrix} \dfrac{\partial}{\partial x} & 0 & 0 & 0 & \frac{\partial}{\partial z} & \frac{\partial}{\partial y} \\ 0 & \frac{\partial}{\partial y} & 0 & \frac{\partial}{\partial z} & 0 & \frac{\partial}{\partial x} \\ 0 & 0 & \frac{\partial}{\partial z} & \frac{\partial}{\partial y} & \frac{\partial}{\partial x} & 0 \end{bmatrix}^{\mathrm{T}}, \\ \mathbf{C} &= \begin{bmatrix} C_{11} & \ldots & C_{16} \\ \vdots & \ddots & \vdots \\ C_{16} & \cdots & C_{66} \end{bmatrix}, \end{aligned} \tag{2}$$

equation (1) represents the velocity-stress elastic wave equations, where $\mathbf{C}$ is the Hooke tensor matrix. When $\boldsymbol{\sigma}$ reduces to the scalar pressure p, $\mathbf{C}$ reduces to $K = \rho v^2$, and $\mathbf{E} = \begin{bmatrix} \partial/\partial x & \partial/\partial y & \partial/\partial z \end{bmatrix}^{\mathrm{T}}$, equation (1) represents the first-order velocity-stress acoustic wave equations. We adopt the convolutional perfectly matched layer (CPML) (Komatitsch and Martin, 2007; Chen et al., 2014) around the model to avoid the boundary reflection.

Next, we provide several explanations and assumptions related to the remainder of this paper. We first briefly introduce the data accessing of the CPU. Generally, the memory types include the processor register, cache memory, random-access memory (RAM), and hard disk. The processor register is the fastest computer memory technology with the least amount of storage space. Cache memory is the second fastest and second smallest amount of memory available in the memory hierarchy. RAM is much larger than the cache memory, with typically 10–100 times slower R/W (reading and writing) latencies. The hard disk is an extension to the memory hierarchy that has much larger storage space but much greater latency. When data are requested, the CPU first accesses the data from the registers; if it fails, it then accesses the data from the cache memory or RAM. A cache hit means the required data are already prestored in registers or cache memory; a high cache hit ratio improves the speed at which the CPU can access data, thereby improving the computational efficiency. Additionally, the code is hybrid programmed by exploiting the hybrid message passing interface (MPI) and open multiprocessing (OpenMP). To make better use of the shared memory multithreaded features of OpenMP, we suggest setting the number of subdomains equal to the number of computer nodes; this point is explained in the next subsection about the kernel of forward modeling.

For the 3D models used in this paper, we define the one-dimensional array index written as $ii = (i_z - 1)N_yN_x + (i_y - 1)N_x + (i_x - 1)$, where N_x, N_y, N_z are the numbers of grid cells in the x-, y-, z-directions, respectively, $i_x = 1, 2, \cdots, N_x$, $i_y = 1, 2, \cdots, N_y$, and $i_z = 1, 2, \cdots, N_z$. We define the dimension in which the data are continuously saved as the fastest dimension. Thus, x is the fastest dimension, y is the second fastest dimension, and z is the slowest dimension. We define D as the number of subdomains for the 1D decomposition; D_x and D_y represent the numbers of subdomains in the $x-$ and $y-$directions, respectively. We always set $D = D_xD_y$ in this paper.

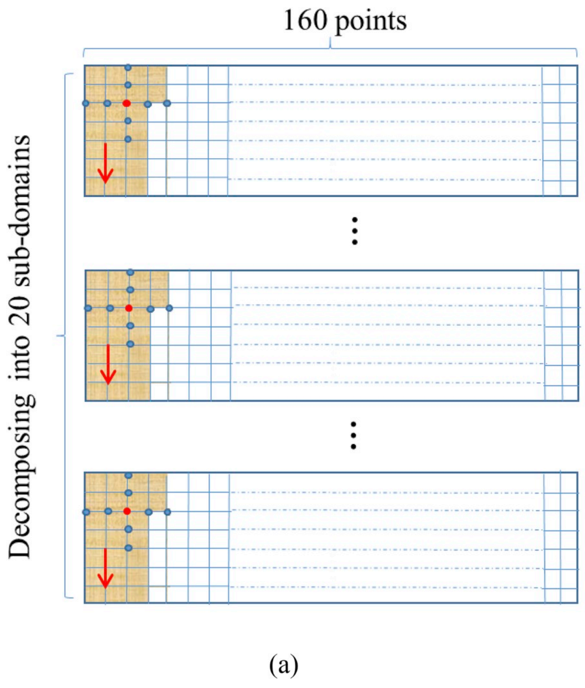
2.2. Optimized decomposition strategy

When the model is too large to be calculated by a single PC or workstation, domain decomposition is necessary. We first theoretically analyze the impact of the domain decomposition direction on the cache hit ratio and then discuss the effect of the domain decomposition dimension on the communication among nodes.

2.2.1. Domain decomposition direction

For simplicity, we illustrate the optimal decomposition direction with a two-dimensional (2D) model sampled into 160×160 cells.

Decomposing this 2D model evenly into 20 subdomains gives two scenarios: perpendicular to the fastest dimension (Fig. 1a) and perpendicular to the slowest dimension (Fig. 1b). The red arrows represent the direction in which the data are stored continuously; this direction is called the fastest dimension. If we calculate the wavefield at the 'red point' using a classical nine-point difference scheme, we need the wavefield at eight 'blue points' around the 'red point'. Addressing these eight 'blue points' in each subdomain requires traversing 35 and 643 data points (the area covered by the yellow background) in Fig. 1a and b, respectively. When the 256-bit advanced vector extensions (AVX) is adopted, which means that eight single-precision floating-point numbers can be simultaneously stored in the registers, data addressing can be finished within 5 and 81 instruction cycles in Fig. 1a and b, respectively. Obviously, the decomposition direction in Fig. 1a can



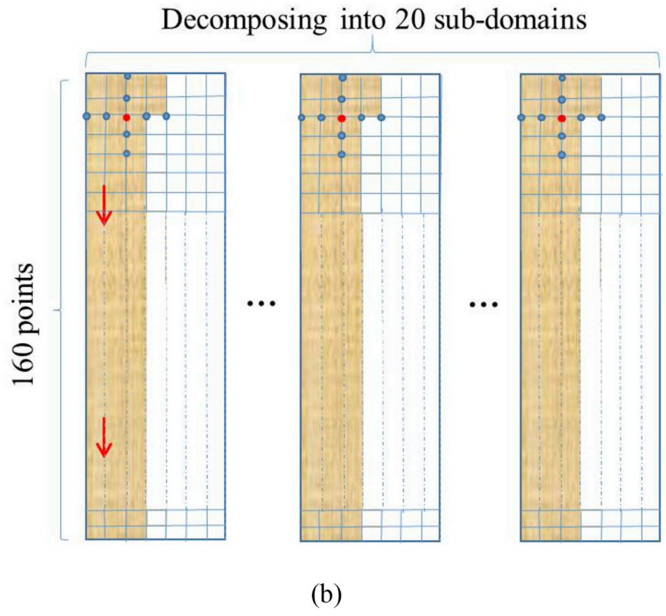


Fig. 1. Illustration of two model decomposition directions: (a) perpendicular to the fast dimension, (b) perpendicular to the slow dimension.

dramatically increase the cache hit ratio and improve the calculation efficiency. For the 3D models defined in the last subsection, decomposing the domain perpendicular to the x-direction is the best choice, and decomposing perpendicular to the z-direction is the worst choice.

2.2.2. Domain decomposition dimension

The domain decomposition dimension mainly affects the communications between nodes. Since the speed of data transmission in the network is slower than that of the numerical calculation in the CPU, too many communications among the nodes will slow the overall computational efficiency. Our goal is to mitigate the communication overhead by a reasonable domain decomposition dimension. We consider two cases of 1D (perpendicular to the *x*-direction) and 2D (perpendicular to the *x*- and *y*-directions) decompositions as shown in Fig. 2, where the blue and red regions represent the subdomains that need the most and least communications, respectively.

Considering a 3D model without the PML boundary, the data size of the communications at each time step in the network (denoted as G_{trans}) for the 1D and 2D decomposition cases are given by

$$G_{trans} = 2DLN_{comp}N_zN_y, \tag{3}$$

$$G_{trans} = 2DLN_{comp}N_z\left(\frac{N_x}{D_x} + \frac{N_y}{D_y}\right), \tag{4}$$

respectively, where L is the FD stencil, which is equal to half of the order of the spatial difference accuracy, and N_{comp} represents the number of wavefield components that need to communicate. Detailed information about N_{comp} is shown in Table 1. Next, we quantify the difference in communication caused by the different decomposition dimensions. We decompose a fixed-size model into different numbers of subdomains; detailed information is listed in Table 2. We attempt to make each node bear the same amount of computation. Since we assume $N_x = N_y$ in Table 2, for load balance considerations, we set $D_x = D_y$. By calculating equations (2) and (3) with $L = 10$, we obtain the histograms in Fig. 3a and b, respectively, which refer the acoustic and elastic cases. We then fix the model as shown in red in Table 2 but change the FD stencil from 1 to 10. Fig. 3c and d shows how G_{trans} changes with increasing FD accuracy for the acoustic and elastic cases, respectively. The 2D decomposition introduces fewer communications, and the more nodes that are used or the higher the order of FD is adopted, the greater the advantages of the 2D decomposition are. Additionally, the 2D decomposition relieves a greater communication burden in the elastic modeling than in the acoustic modeling.

We did not consider the 3D decomposition scheme in this paper, mainly because 3D decomposition is more prone to suffering load imbalances caused by the storage of seismic records. Assuming the 3D model is decomposed into 64 subdomains and that the receivers are located at every grid point in the *x*-*y* plane, the seismic record can be expressed as $G_{record} = N_x \times N_y \times N_t$, where N_t represents the number of time steps. For both the 1D decomposition (64 subdomains perpendicular to the *x*-direction) and the 2D decomposition (8 subdomains perpendicular to *x*- and y-directions), every node contains the receivers and needs to store the same seismic record of $G_{record}/64$; for the 3D decomposition (4 subdomains perpendicular to the *x*-, *y*- and *z*-directions), 16 nodes contain the receivers (these 16 nodes store the same seismic record of $G_{record}/16$), but the other nodes do not to store seismic records. Since the hard disk writing speed is significantly slower than the CPU computing speed, and the overall execution time depends on the slowest node, the nodes that need to store seismic records in the 3D decomposition are likely to decrease the overall computational efficiency.

3. Performance evaluation

We have theoretically analyzed the effects of domain decomposition

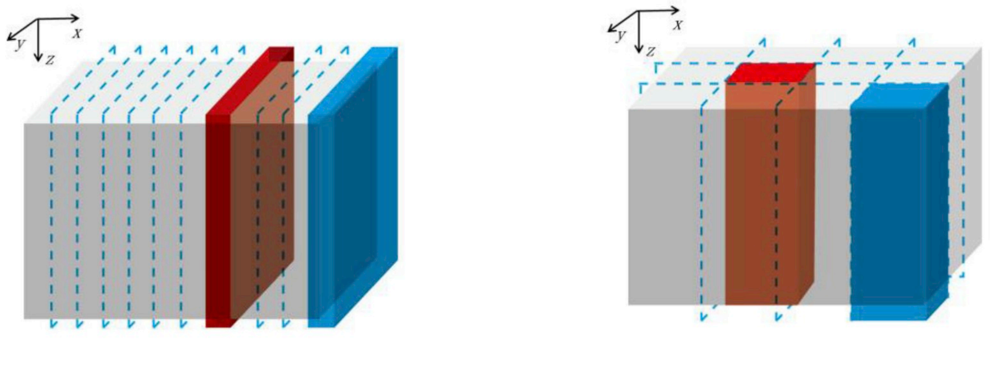


Fig. 2. Schematic illustration of (a) 1D domain decomposition and (b) 2D domain decomposition. The blue and red regions represent the subdomains that require the most and least communications, respectively. (For interpretation of the references to colour in this figure legend, the reader is referred to the Web version of this article.)

Table 1
The wavefield components that need to communicate.

	Acoustic wave equation	Elastic wave equation
1D Decomposition	x-direction: p v_x	x-direction: σ_{xx} σ_{xz} σ_{xy} v_x v_y v_z
2D Decomposition	x-direction: p v_x y-direction: p v_y	x-direction: σ_{xx} σ_{xz} σ_{xy} v_x v_y v_z y-direction: σ_{yy} σ_{xy} σ_{yz} v_x v_y v_z

Table 2
Seven groups of models with different subdomains.

$N_x = N_y = N_z$	$D = D_xD_y$	$D_x = D_y$
1024	4	2
1024	9	3
1024	16	4
1024	25	5
1024	36	6
1024	49	7
1024	64	8

on the cache hit and communication. In this section, we evaluate the computational performance by implementing 3D acoustic simulations on a homogeneous model with $1024 \times 1024 \times 1024$ grid points. We consider three scenarios: 1D decomposition perpendicular to the fastest dimension (denoted as 1D_f) and the slowest dimension (denoted as 1D_s) and a 2D optimized decomposition scheme (perpendicular to the fastest and the second fastest dimensions). The purpose of comparing 1D_f and 1D_s is to illustrate the effects of the decomposition direction on the computational efficiency, and the comparison of the 1D_f and 2D decomposition methods illustrates the effects of the decomposition dimension.

3.1. Computation time

Tables 3 and 4 show the computation times of a 500-timestep simulation. In Table 3, the FD stencil is $L = 5$, and the number of nodes increases from 1 to 64 with an exponential growth rate. A comparison of the computation times corresponding to 1D_f and 1D_s shows that the direction of the domain decomposition has a considerable impact on the computational efficiency. For the 64-node case, the computational efficiency of 1D_f is almost three times that of 1D_s. Then, we fix the nodes at 64 but increase the FD stencil from 1 to 10; the computation time is shown in Table 4. Not surprisingly, the calculation time increases with the increase of the FD accuracy. The calculation time of 1D_s is much longer than those of the other two schemes. The calculation time for 1D_f increases rapidly after $L = 9$. We believe the possible reason is that as L increases, G_{trans} also increases, as shown in Fig. 3b, but after $L = 9$, the communication of 1D_f cannot be completely hidden by the 'overlap communication and computation' method (to be introduced in the next section). According to our tests, for the 2D decomposition, this surge occurs when $L = 13$.

3.2. Speedup ratio and strong scaling efficiency

Using the data shown in Fig. 4a, we further evaluate the parallel performance using the metrics of the speedup ratio S_p and the strong scaling efficiency E_{ss}, which are defined as

$$S_p = \frac{t_1}{t_N}, \tag{5}$$

$$E_{ss} = \frac{t_1}{N \times t_N} \times 100\%, \tag{6}$$

where t_1 and t_N represent the calculation times of the simulation calculated by one node and N nodes, respectively. Ideally, the speedup ratio equals N, and E_{ss} equals 100%.

Fig. 4a, b and 4c show the speedup ratios and strong scaling efficiencies for the 1D_s, 1D_f, and 2D decomposition methods, respectively. The speedup ratio of the 1D_s decomposition method does not improve significantly after the number of nodes exceeds 25. The speedup ratio is only 14.47 for 64 nodes, which is unacceptable. A comparison of Fig. 4b and c shows no conspicuous differences between them with fewer than 16 nodes, but the advantages of the 2D decomposition gradually emerge when more nodes are used. When 64 nodes are used, the speedup ratios of the 1D_f and 2D decompositions are 41.22 and 52.88, respectively, and the strong scaling efficiencies are 64% and 82%, respectively. Thus, the 2D decomposition is more suitable for fine-grained parallelism.

3.3. Memory usage

The direction of the domain decomposition has no effect on memory usage, which means that the memory usages of the 1D_f and 1D_s decompositions are exactly the same. Thus, we only compare the 1D and 2D decomposition cases. We define the memory usage efficiency as

$$E_M = \frac{M_1}{N \times M_N} \times 100\% \tag{7}$$

where the memory usage of one node is M_1, and that of N nodes is M_N. As

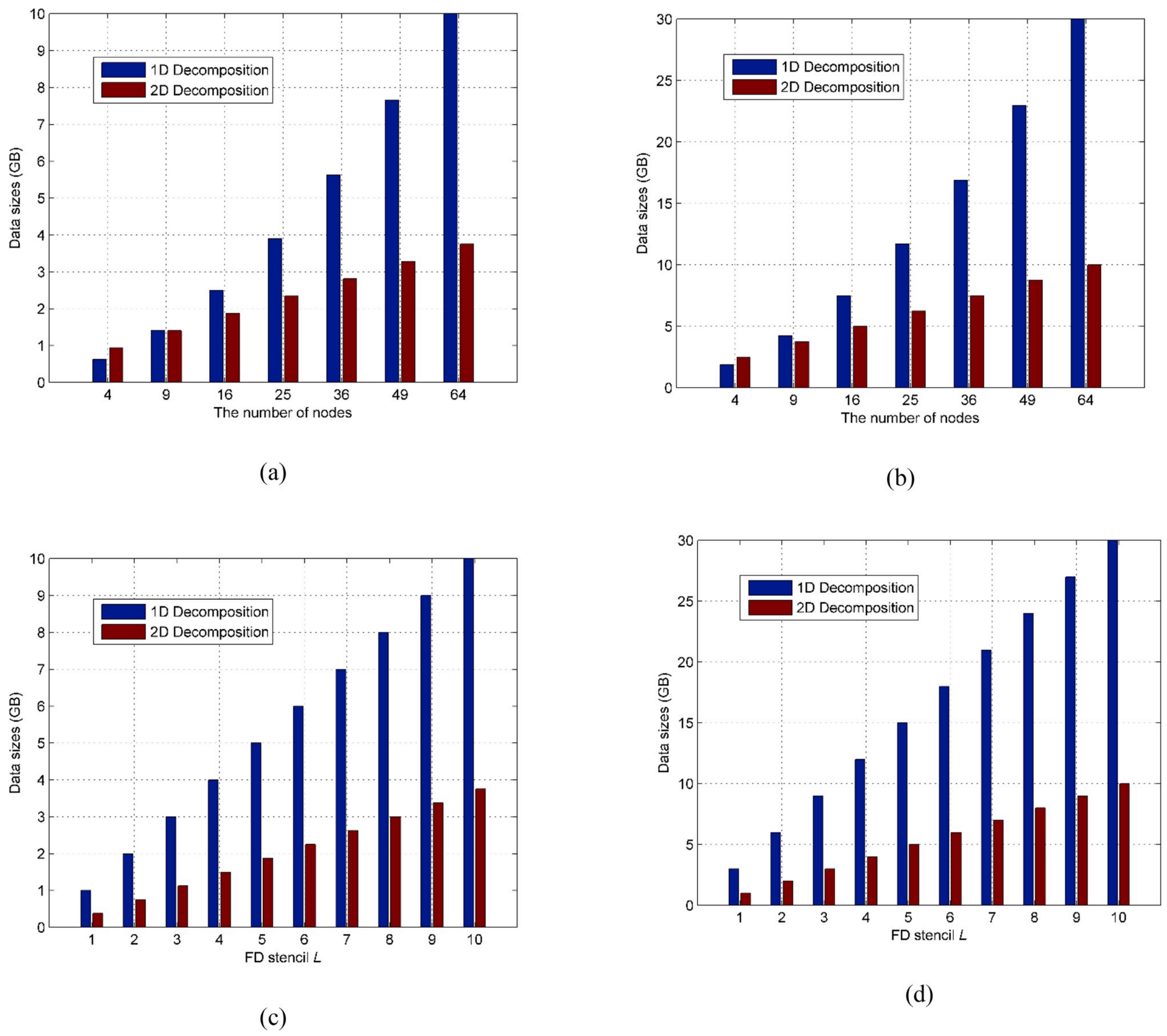


Fig. 3. Changes in G_{trans} with the number of nodes for a fixed FD stencil $L = 10$ for (a) acoustic and (b) elastic cases. The relationship of the communication burden with the FD stencil L for a fixed-size model as shown in the red figures in Table 1 for (c) acoustic and (d) elastic cases. (For interpretation of the references to colour in this figure legend, the reader is referred to the Web version of this article.)

Table 3
Computation times (in second) of a 500-timestep simulation for 3 decomposition schemes when the FD stencil $L = 5$, and the number of nodes increases from 1 to 64.

Nodes	1	4	9	16	25	36	49	64
1D_s	10100	2961	1603	1013	756	721	702	698
1D_f	10100	2738	1222	732	495	374	290	245
2D	10100	2730	1214	705	470	335	249	191

shown in Fig. 5, with an increasing number of compute nodes, the memory usage efficiency decreases for both the 1D and 2D decompositions. However, the 2D decomposition always performs better than the 1D decomposition. In fact, the MPI allocates some memory for communication buffers to store the data to be transmitted. The curves in Fig. 5 indicate that the 2D decomposition leads to fewer transmitted data, which echoes the conclusions drawn from Fig. 3.

4. Description of the package

Our package aims to provide a set of 3D parallel simulation schemes based on the optimized decomposition strategies for propagating

Table 4
Computation times (in second) of a 500-timestep simulation for 3 decomposition schemes when the number of nodes is 64, and the FD stencil increases from 1 to 10.

FD stencil	1	2	3	4	5	6	7	8	9	10
1D_s	358	381	409	441	479	513	551	599	647	698
1D_f	132	133	138	145	153	164	176	199	269	345
2D	132	133	137	142	148	155	163	171	180	191

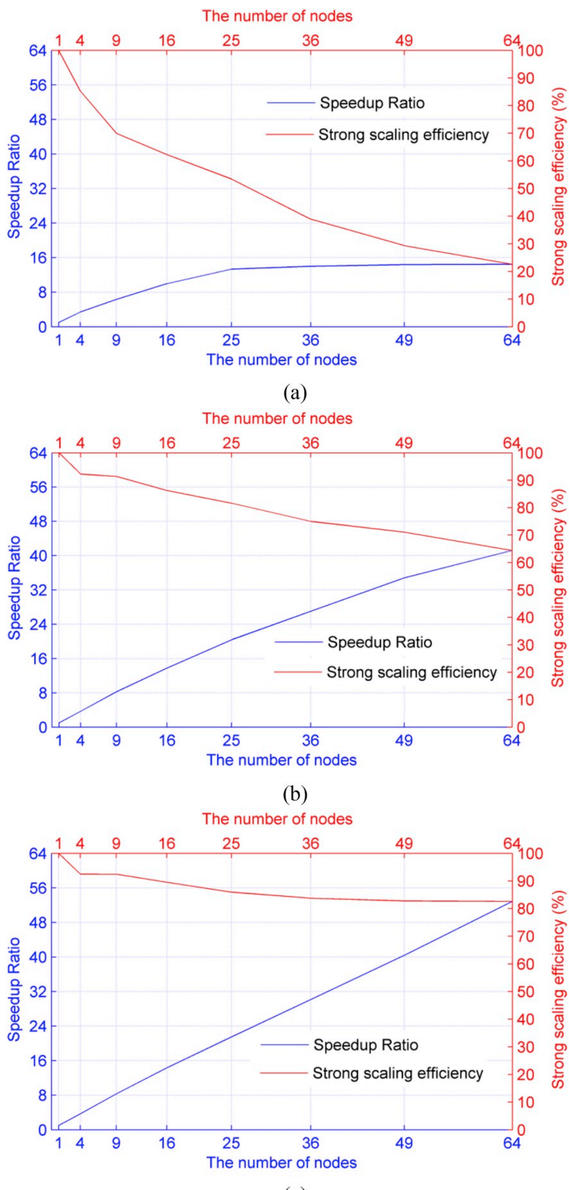


Fig. 4. Speedup ratios and strong scaling efficiencies corresponding to the (a) 1D_s, (b) 1D_f, and (c) 2D decomposition methods.

acoustic and elastic waves. In this section, we mainly outline the architecture of our code package and underline several program optimization schemes. Our package includes two parts: the **optimized version** is based on the proposed optimized 2D decomposition scheme, and the **verification version** is designed to verify the efficiency of the optimized decomposition scheme, in which the codes are based on the 1D_f, 1D_s, and 2D optimized schemes.

We take the optimized scheme-based 3D acoustic wave modeling code **Acoustic_de2D** as an example to illustrate the architecture of our code. The codes can be roughly separated into three components: preprocessing, the kernel of forward modeling, and postprocessing. As shown in Fig. 6, each component plays an indispensable role in parallel computing. A brief description of each component is given below.

4.1. *Preprocessing*

The preprocessing consists of two parts. The first part is the model parameterization, in which we input the parameters of the model and its geometry, define the number of subdomains, and define the size of the model and the number of available compute nodes. The second part is designed to decompose the velocity and density model and to allow each node to read the velocity and density of the submodel to be simulated. Note that for load balancing, the object being decomposed is the model containing the PML region instead of the original velocity and density model.

4.2. *Kernel of forward modeling*

After the preprocessing, each node will perform the wavefield modeling on its corresponding subdomains. The package utilizes three methods to address the intensive computation problems. The first is the optimized domain decomposition strategy. The second method is to exploit hybrid MPI and OpenMP programming, which means that the domain is decomposed according to the number of compute nodes, the parallelism among the nodes is completed by MPI, and the parallelism within a node is completed by OpenMP. Due to the shared-memory multithreaded features of OpenMP, the communication cost within a node can be effectively removed (Chorley and Walker, 2010). The third method is communication overlap. In the context of multinode seismic modeling, MPI communications take up a larger fraction of the application execution time (Abdelkhalek et al., 2012). In addition to reducing the transmitted data by the aforementioned 2D decomposition method, the natural problem of MPI communication can be further overcome by leveraging parallelism between the processing and communication units to overlap the communication and computation. This classical ‘overlap communication and computation’ method can be outlined in the following steps:

(1) Calculate the communication areas that need be exchanged with the neighboring subdomains.
(2) Extract the MPI buffers from the main arrays.
(3) Launch asynchronous repeated MPI ‘send’ and ‘receive’ commands.
(4) Exchange the data of the communication areas and calculate the wavefield of the internal areas. Note that these two processes are implemented simultaneously.
(5) Use wait operators to ensure that the two processes in (4) have been completed.
(6) Update the main arrays based on the content of the MPI buffers received.
(7) Repeat these operations at every time step.

4.3. *Postprocessing*

Each node will output the corresponding seismic subrecord when the forward modeling is completed. In the postprocessing, we need to splice the subrecords that are generated by all of the nodes into a complete seismic record. The postprocessing also includes terminating the MPI process and releasing memory.

5. Application examples

In this section, three examples, including 3D acoustic and elastic forward modeling and 3D acoustic RTM, are performed to verify the feasibility and efficiency of the proposed optimized parallel scheme. These numerical tests are performed on the Tianhe-1A supercomputer, which is provided by the National Supercomputing Center in Tianjin, China.

The first numerical example simulates acoustic wave propagation in a realistic model of the Bohai Bay Basin shown in Fig. 7. This model was developed using a variety of information, including the subsurface geological and near-surface characteristics, prestack depth migration profiles, and well log data. Thus, this model fully reflects the tectonic, sedimentary and seismic reflection characteristics of the Bohai Bay Basin. This complex realistic model contains $1517 \times 1601 \times 2402$ grid

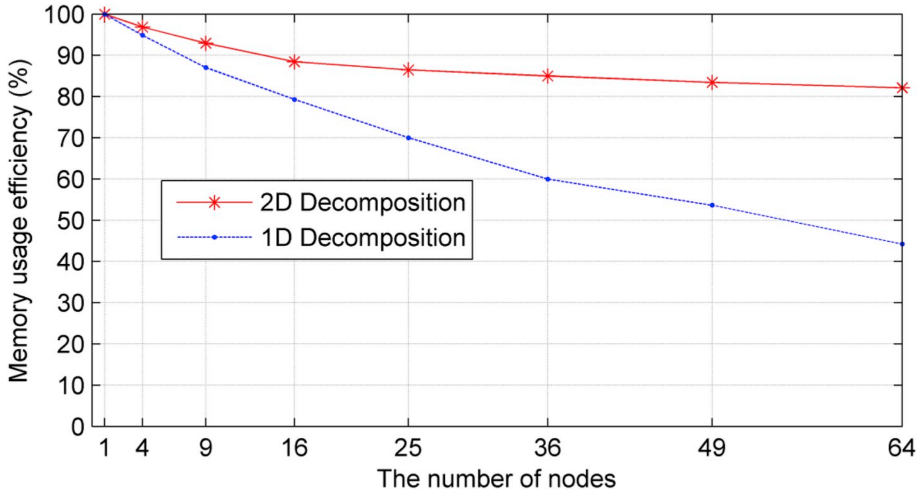


Fig. 5. Memory usage efficiency with the different numbers of compute nodes. The blue and red lines represent the memory usage efficiencies corresponding to the 1D and 2D decompositions, respectively. (For interpretation of the references to colour in this figure legend, the reader is referred to the Web version of this article.)

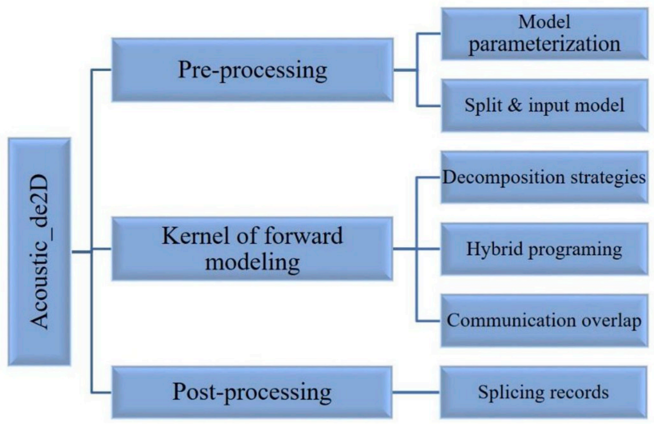


Fig. 6. The architecture of the 3D acoustic wave modeling code **Acoustic_de2D.**

points (the data size of the model is 22 GB), and the velocities are distributed over a wide range from 1700 m/s to 7000 m/s. The grid spacings in the x-, y-, and z-directions are 6.25 m, 6.25 m, and 5 m, respectively. The simulation time duration is set to 5.625 s with a time step of 0.45 ms; thus, there are 12500 time steps. A Ricker wavelet excitation source with a dominant frequency of 20 Hz is located at (5000 m, 4700 m, 50 m), and the receivers are located at every grid point on the x-y plane at a depth of 50 m. We employ 64 nodes to perform the simulations. Fig. 8 shows a common-shot seismic record, where the top shows a horizontal slice at 2.25 s, the inline profile is at 5 km in the crossline direction, and the crossline profile is at 4.75 km in the inline direction. For the optimized 2D domain decomposition method (decomposing the model into 8 subdomains in the x- and y-directions, respectively), the total wall-clock time is 6231 s. With the same simulation time, the 1D_f (decomposing the model into 64 subdomains in the x-direction) and 1D_s (decomposing the model into 64 subdomains in the z-direction)-based codes can complete the calculation in 9800 and 3400 time steps, respectively.

In the second numerical example, we simulate elastic wave propagation in a classical overthrust model. Fig. 9 shows the P-wave velocity model, which contains $801 \times 801 \times 187$ grid points with a uniform grid spacing of 10 m. We generate S-wave velocities using the relation $v_s = v_p/1.73$. The seismic record lasts 2.8 s with a time step of 0.7 ms; thus, there are 4000 time steps. A Ricker wavelet excitation source with a dominant frequency of 20 Hz is located at (4000 m, 4000 m, 30 m), and the receivers are located at every grid point on the x-y plane at a depth of

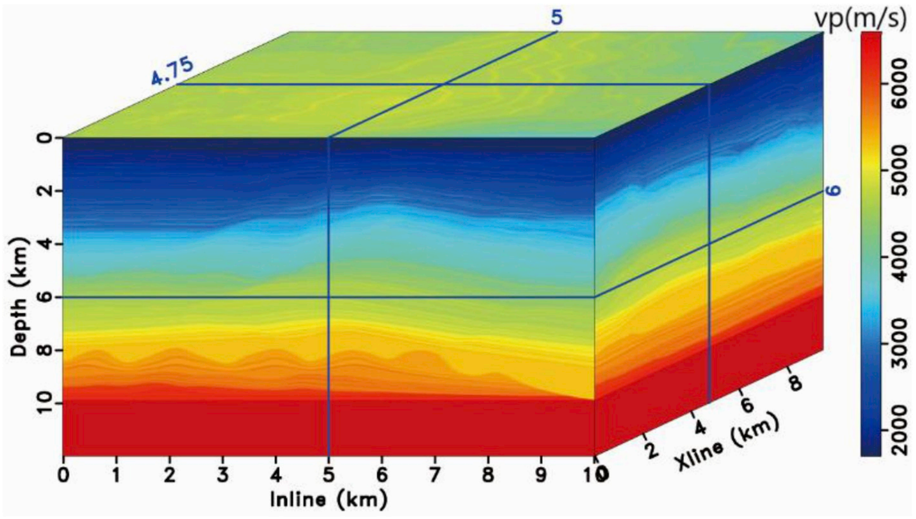


Fig. 7. P-wave velocity model of the Bohai Bay Basin.

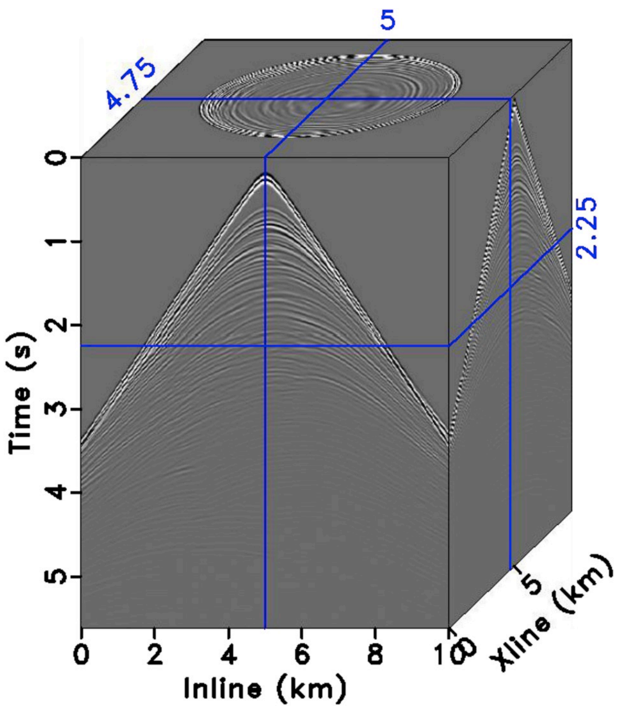


Fig. 8. A 3D acoustic common-shot gather computed by 64 nodes on the Tianhe-1A supercomputer cluster.

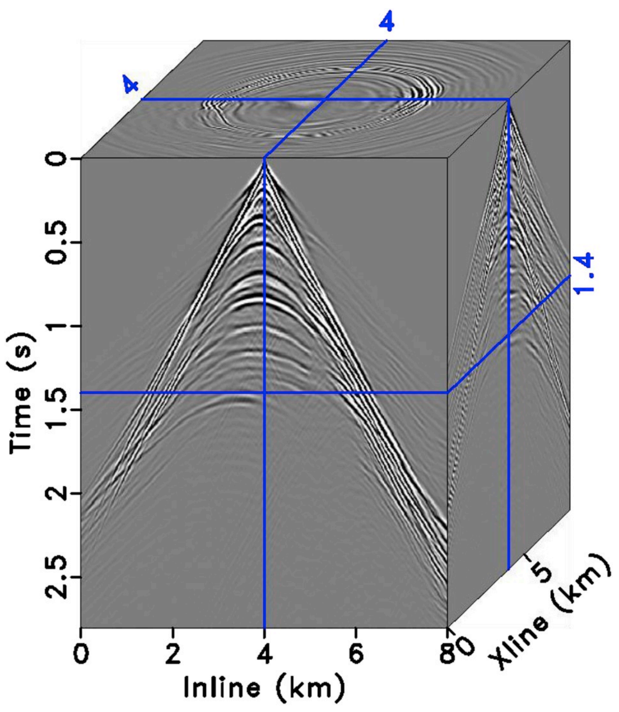


Fig. 10. A 3D elastic common-shot gather computed by 64 nodes on the Tianhe-1A supercomputer cluster.

30 m. We use 64 compute nodes to perform the simulation. Fig. 10 displays a common-shot gather of v_x, where the top shows a horizontal slice at 1.4 s, the inline profile is at 4 km in the crossline direction, and the crossline profile is at 4 km in the inline direction. The total wall-clock times for the optimized 2D decomposition, 1D_f, and 1D_s schemes are 987 s, 1513 s, and 3436 s, respectively.

It is straightforward to use the optimized parallel modeling scheme as the forward engine for performing 3D acoustic RTM. In the third example, we conduct 3D RTM on a truncated overthrust model. Fig. 11 shows the velocity model, which contains 550 × 200 × 180 grid points with a uniform sampling interval of 10 m in the x-, y-, and z-directions. The model is decomposed into 11 equal subdomains in the x-direction (the fastest dimension) and 4 equal subdomains in the y-direction (the second fastest dimension). In the observation system, 28 × 10 sources are distributed in the x- and y-directions, respectively. The time step is 0.6 ms, and the record time is 2.1 s. The point sources are excited by a Ricker wavelet with a dominant frequency of 30 Hz. Fig. 12 shows the migrated image with the normalized crosscorrelation imaging conditions, where the inline profile shows the image at 0.6 km in the crossline direction, and the crossline profile shows the image at 1.2 km in the inline direction. The wall-clock time of the migration is 25148 s on 44 compute nodes.

All three examples demonstrate the ability of the optimized decomposition strategies in handling large-scale simulations. Additionally, due to the fine speedup ratio and strong scaling efficiency of the proposed 2D domain decomposition scheme, better computation times will likely be attained when more compute nodes are employed.

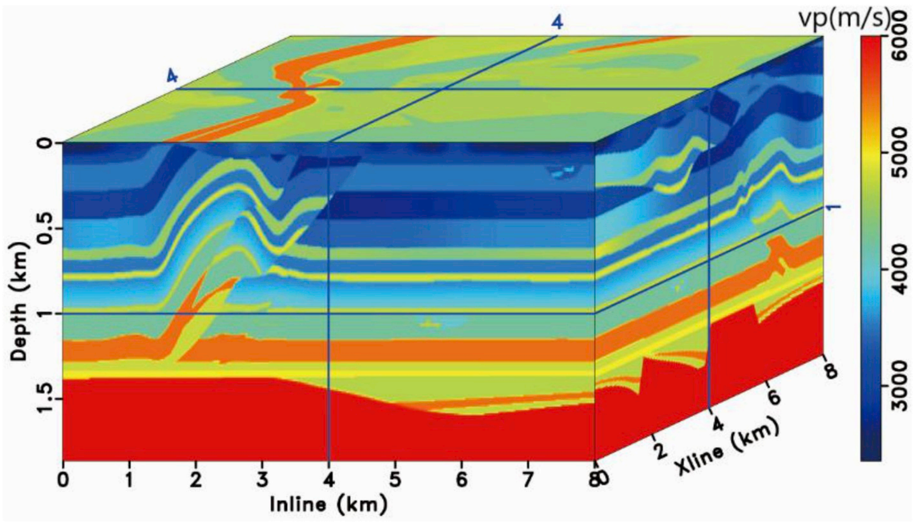


Fig. 9. P-wave velocity of the overthrust model.

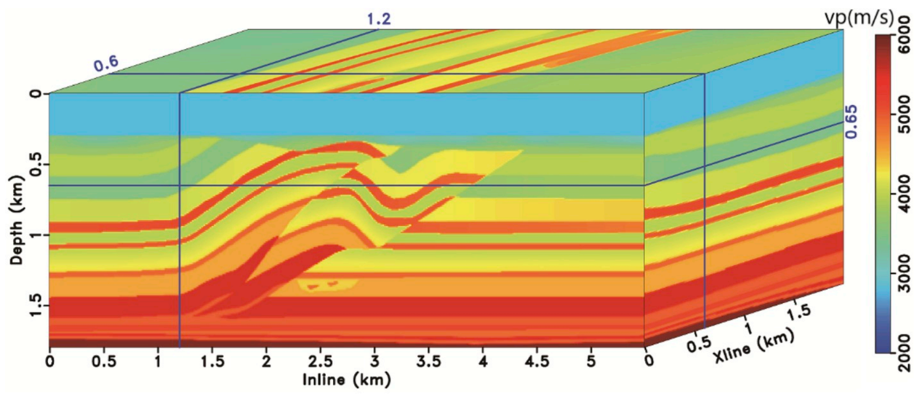


Fig. 11. P-wave velocity of the truncated overthrust model.

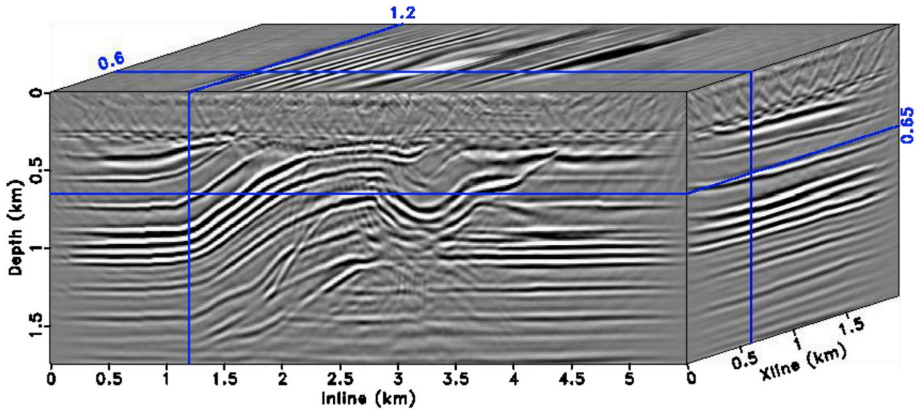


Fig. 12. Migrated image obtained using the 3D acoustic reverse time migration method developed using our optimized parallel modeling scheme.

6. Conclusions

We systematically analyzed the impact of the domain decomposition direction on the cache hit ratio and the effect of the domain decomposition dimension on the communication among nodes. The theoretical analysis showed that the 2D decomposition scheme that decomposes the domains perpendicular to the fastest and second fastest dimensions is a better choice in most cases. Four performance metrics confirm our theoretical analysis and the efficiency of the proposed decomposition scheme. Three examples further demonstrate the feasibility of the proposed optimized parallel scheme for large parallel clusters. The optimized decomposition scheme is applicable to the traditional or other time-space domain high-order finite difference methods.

7. Computer code availiability

The package is available from GitHub at https://github.com/Geo-reader/ODD.

Acknowledgments

We thank Prof. Guofa Li for providing the Bohai Bay Basin model. This work is partly supported by National Natural Science Foundation of China (41630314, 41874130), National Science and Technology Program (2016ZX05010001), Major Project of the China National Petroleum Corporation (The Research of Gaussian Beam Migration of Elastic Waves).

Appendix A. Supplementary data

Supplementary data to this article can be found online at https://doi.org/10.1016/j.cageo.2019.06.017.

References

Abdelkhalek, R., Coulaud, O., Coulaud, O., Roman, J., Roman, J., 2012. Fast seismic modeling and reverse time migration on a graphics processing unit cluster. Concurrency Comput. Pract. Ex. 24 (7), 739–750.
Chen, H., Zhou, H., Li, Y., 2014. Application of unsplit convolutional perfectly matched layer for scalar arbitrarily wide-angle wave equation. Geophysics 79 (6), T313–T321.
Chen, H., Zhou, H., Zhang, Q., Chen, Y., 2017. Modeling elastic wave propagation using k-space operator-based temporal high-order staggered-grid finite-difference method. IEEE Trans. Geosci. Remote Sens. 55 (2), 801–815.
Chen, Y., 2018. Automatic velocity analysis using high-resolution hyperbolic radon transform. Geophysics 83 (4), A53–A57.
Chen, Y., 2018. Fast waveform detection for microseismic imaging using unsupervised machine learning. Geophys. J. Int. 215 (2), 1185–1199.
Chen, Y., Huang, W., Zhou, Y., Liu, W., Zhang, D., 2018. Plane-wave orthogonal polynomial transform for amplitude-preserving noise attenuation. Geophys. J. Int. 214 (3), 2207–2223.
Chen, Y., Bai, M., Guan, Z., Zhang, Q., Zhang, M., Wang, H., 2019. Five-dimensional seismic data reconstruction using the optimally damped rank-reduction method. Geophys. J. Int. 218 (1), 224–246.
Chorley, M.J., Walker, D.W., 2010. Performance analysis of a hybrid MPI/OpenMP application on multi-core clusters. Journal of Computational Science 1 (3), 168–174.
Chu, C., Stoffa, P., Seif, R., 2009. 3D Seismic Modeling and Reverse-Time Migration with the Parallel Fourier Method Using Non-blocking Collective Communications. SEG Expanded Abstracts, pp. 2677–2681.
Cai, X., Chen, G., Fan, X., Chen, Y., 2019. Least-squares based rectangular-grid cross and rhombus stencils for acoustic wave propagation and reverse time migration. J. Comput. Phys. 392, 335–353.
Dablain, M., 1986. The application of high-order differencing to the scalar wave equation. Geophysics 51 (1), 54–66.
Etgen, J., O'Brien, M.J., 2007. Computational methods for large-scale 3D acoustic finite-difference modeling: a tutorial. Geophysics 72 (5), SM223–SM230.
Fu, H., He, C., Chen, B., et al., 2017. Pflops nonlinear earthquake simulation on Sunway TaihuLight: enabling depiction of 18-Hz and 8-meter scenarios. Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis 1–12.
Fang, J., Zhou, H., Chen, H., Wang, N., Wang, Y., Sun, P., Zhang, J., 2018. Source-independent elastic least-squares reverse time migration. Geophysics 84 (1), S1–S16.

Gao, H., Zhang, J., 2006. Parallel 3-D simulation of seismic wave propagation in heterogeneous anisotropic media: a grid method approach. Geophys. J. Int. 165 (3), 875–888.

Gokhberg, A., Fichtner, A., 2016. Full-waveform inversion on heterogeneous HPC systems. Comput. Geosci. 89, 260–268.

Guo, P., McMechan, G.A., 2017. Sensitivity of 3D 3C synthetic seismograms to anisotropic attenuation and velocity in reservoir models. Geophysics 82 (2), T79–T95.

Guo, P., McMechan, G.A., 2018. Compensating Q effects in viscoelastic media by adjoint-based least-squares reverse time migration. Geophysics 83 (2), S151–S172.

Guan, Z., Niu, F., 2017. An investigation on slowness-weighted CCP stacking and its application to receiver function imaging. Geophys. Res. Lett. 44, 6030–6038.

Guan, Z., Niu, F., 2018. Using fast marching Eikonal solver to compute 3-D PDS traveltime for deep receiver-function imaging. J. Geophys. Res.: Solid Earth 123, 9049–9062.

Huang, C., Dong, L.G., 2009. High-order finite-difference method in seismic wave simulation with variable grids and local time-steps (in Chinese). Chin. J. Geophys. 52, 176–187.

Huang, C., Dong, L.G., 2009. Staggered-grid high-order finite-difference method in elastic wave simulation with variable grids and local time-steps (in Chinese). Chin. J. Geophys. 52, 1324–1333.

Komatitsch, D., Martin, R., 2007. An unsplit convolutional perfectly matched layer improved at grazing incidence for the seismic wave equation. Geophysics 72 (5), SM155–SM167.

Lindstrom, P., Chen, P., Lee, E.,J., 2016. Reducing disk storage of full-3D seismic waveform tomography (F3DT) through lossy online compression. Comput. Geosci. 93, 45–54.

Li, S., Liu, B., Ren, Y., Chen, Y., Yang, S., Wang, Y., Jiang, P., 2019. Deep Learning Inversion of Seismic Data arXiv preprint arXiv:1901.07733.

Olsen, K.B., Archuleta, R.J., Matarese, J.R., 1995. Three-dimensional simulation of a magnitude 7.75 earthquake on the San Andreas fault. Science 270 (5242), 1628–1632.

O'Brien, G.S., Bean, C.J., Tapamo, H., 2009. Dispersion analysis and computational efficiency of elastic lattice methods for seismic wave propagation. Comput. Geosci. 35, 1768–1775.

Qu, S., Verschuur, E., Chen, Y., 2019. Full-waveform inversion and joint migration inversion with an automatic directional total variation constraint. Geophysics 84 (2), R175–R183.

Raknes, E.B., Arntsen, B., 2016. Challenges and solutions for performing 3D time-domain elastic full-waveform inversion. Lead. Edge 36 (1), 88–93.

Ren, Z., Kalscheuer, T., Greenhalgh, S., Maurer, H., 2014. A finite-element-based domain-decomposition approach for plane wave 3D electromagnetic modeling. Geophysics 79 (6), E255–E268.

Rubio, F., Hanzich, M., Farrés, A., De La Puente, J., Cela, J.M., 2014. Finite-difference staggered grids in GPUs for anisotropic elastic wave propagation simulation. Comput. Geosci. 70, 181–189.

Sheen, D.-H., Tuncay, K., Baag, C.-E., Ortoleva, P.J., 2006. Parallel implementation of a velocity-stress staggered-grid finite-difference method for 2D poroelastic wave propagation. Comput. Geosci. 32, 1182–1191.

Said, I., Fortin, P., Lamotte, J., Calandra, H., 2017. Leveraging the accelerated processing units for seismic imaging: a performance and power efficiency comparison against CPUs and GPUs. Int. J. High Perform. Comput. Appl. 1, 109434201769656.

Shukla, K., Carcione, J.M., Pestana, R.C., Jaiswal, P., Özdenvar, T., 2019. Modeling the wave propagation in viscoacoustic media: an efficient spectral approach in time and space domain. Comput. Geosci. 126, 31–40.

Virieux, J., Operto, S., 2009. An overview of full-waveform inversion in exploration geophysics. Geophysics 74 (6), WCC127–WCC152.

Wang, Y., Zhou, H., Chen, H., Chen, Y., 2017. Adaptive stabilization for Q-compensated reverse time migration. Geophysics 83 (1), S15–S32.

Wang, N., Zhou, H., Chen, H., Xia, M., Wang, S., Fang, J., Sun, P., 2018. A constant fractional-order viscoelastic wave equation and its numerical simulation scheme. Geophysics 83 (1), T39–T48.

Wang, Y., Zhou, H., Zhao, X., Zhang, Q., Zhao, P., Yu, X., Chen, Y., 2018. CuQ-RTM: a CUDA-based code package for stable and efficient Q-compensated RTM. Geophysics 84 (1), 1–69.

Weiss, R.M., Shragge, J., 2013. Solving 3D anisotropic elastic wave equations on parallel GPU devices. Geophysics 78 (2), F7–F15.

Yang, P., Gao, J., Wang, B., 2014. RTM using effective boundary saving: a staggered grid GPU implementation. Comput. Geosci. 68, 64–72.

Zhao, Q., Du, Q., Gong, X., Chen, Y., 2018. Signal-preserving erratic noise attenuation viaiterative robust sparsity-promoting filter. IEEE Trans. Geosci. Remote Sens. 56, 3547–3560.

Zhang, Q., Zhou, H., Li, Q., Chen, H., Wang, J., 2015. Robust source-independent elastic full-waveform inversion in the time domain. Geophysics 81 (2), R13–R28.

Zhang, Q., Mao, W., Zhou, H., Zhang, H., Chen, Y., 2018. Hybrid-domain simultaneous-source full waveform inversion without crosstalk noise. Geophys. J. Int. 215 (3), 1659–1681.

Zhang, Q., Mao, W., Chen, Y., 2018. Attenuating crosstalk noise of simultaneous-source least-squares reverse time migration with GPU-based excitation amplitude imaging condition. IEEE Trans. Geosci. Remote Sens. 99, 1–11.

Zhao, X., Zhou, H., Wang, Y., Chen, H., Zhou, Z., Sun, P., Zhang, 2018. A stable approach for Q-compensated viscoelastic reverse time migration using excitation amplitude imaging condition. Geophysics 83 (5), S459–S476.

Zhou, M., Symes, W., 2014. Wave Equation Based Stencil Optimizations on Multi-Core CPU. SEG Expanded Abstracts, pp. 3551–3555.