

Parallelization of Atomic Image Reconstruction from X-ray Fluorescence Holograms with XcalableMP



Atsushi Kubota, Tomohiro Matsushita, and Naohisa Happo

Abstract X-ray fluorescence holography is a three-dimensional middle range local structure analysis method, which can provide three-dimensional atomic images around specific elements within a radius of a few nanometers. Three-dimensional atomic images are reconstructed by applying discrete Fourier transform (DFT) to hologram data. Presently, it takes long time to process this DFT. In this study, the DFT program is parallelized by using a parallel programming language XcalableMP. The DFT process, whose input is 21 holograms data of 179×360 points and output is a three-dimensional atomic image of 192^3 points, is executed on PC cluster which consists of 8 nodes of Intel Xeon X5660 processors and 96 cores in total and we confirmed that the parallelized DFT execution is 94 times faster than the sequential execution.

1 Introduction

X-ray fluorescence holography (XFH) is a three-dimensional middle range local structure analysis method, which can prove 3D atomic images around specific elements within a radius of a few nanometers[4]. Compared to other method such as X-ray diffraction, which has been widely used for structure analysis of crystals and other materials, XFH is more sensitive to atomic fluctuations, and therefore it is useful for characterization of local lattice distortions.

In the XFH method, hologram data are obtained by experiments done at large synchrotron facilities such as SPring-8 and KEK-PF. Three-dimensional atomic images are reconstructed from the obtained holograms by Barton's method[1, 2].

A. Kubota (✉) · N. Happo
Hiroshima City University, Hiroshima, Japan
e-mail: kubota@hiroshima-cu.ac.jp; happo@hiroshima-cu.ac.jp

T. Matsushita
Nara Institute of Science and Technology, Ikoma, Nara, Japan
e-mail: t-matusita@ms.naist.jp

In Barton's method, it takes long time to reconstruct the atomic images from the obtained holograms.

In distributed-memory computing environment such as PC clusters and supercomputers, hybrid parallelization is widely used by describing inter- and intra- node parallelism with Message-Passing Interface (MPI) [6] and OpenMP [7], respectively. However, it is pointed out that MPI programs tend to be complicated and error-prone. Therefore, in order to improve productivity of parallel programming, a variety of parallel programming languages, language extensions, and libraries have been proposed. Some examples of them are High Performance Fortran (HPF) [5], CoArray adopted in Fortran2008, UPC [9], which is based on C language, and XcalableMP [10], in which distribution of data and parallelization of loops are specified by directives. New parallel programming languages such as X10 [8] and Chapel [3] are also proposed.

We adopted a hybrid parallel programming approach, in which inter- and intra-node parallelism are described in XcalableMP and OpenMP, respectively, to parallelize the existing atomic image reconstruction program by Barton's method written in C language because it can be parallelized with small amount of modification.

In the rest of this chapter, X-ray fluorescence holography and reconstruction of atomic images are explained in Sect. 2. Parallelization of atomic image reconstruction is explained in Sect. 3 and its performance results are shown in Sect. 4. Finally, concluding remarks are given in Sect. 5

2 X-ray Fluorescence Holography

There are two modes in XFH, namely the normal mode and the inverse mode. In this chapter, we focus on the inverse mode because it is mainly used in experiments of XFH recently. Please refer to the literatures such as [4] in details.

In the inverse mode, the angle of a sample material to the incident X-ray is varied as shown in Fig. 1 and intensity of X-ray fluorescence emitted from atoms in the sample is measured by the detector.

As shown in Fig. 2, the incident X-ray approaching atom A and another incident X-ray also approaching atom A after scattered by atom B form a constant wave of X-ray around atom A. The pattern of the constant wave from atom A varies according to the angle of the incident X-ray and resulted in the variation of the intensity of the X-ray fluorescence from atom A. Experimental data obtained by measuring the intensity of X-ray fluorescence make a hologram of the atomic image.

Incident X-ray wave approaching atom A directly corresponds to reference wave of ordinary hologram and incident X-ray wave approaching atom A after scattered by atom B corresponds to object wave.

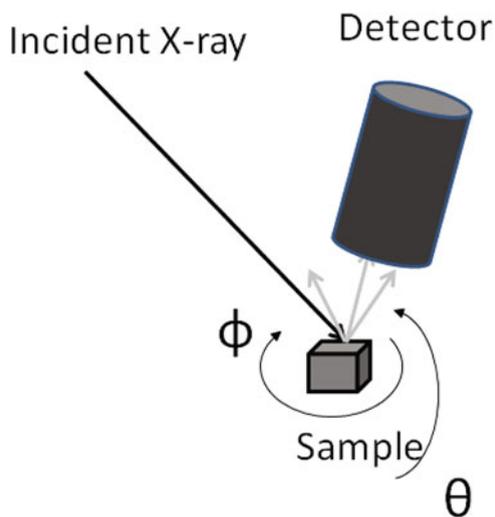


Fig. 1 Inverse mode of XFH

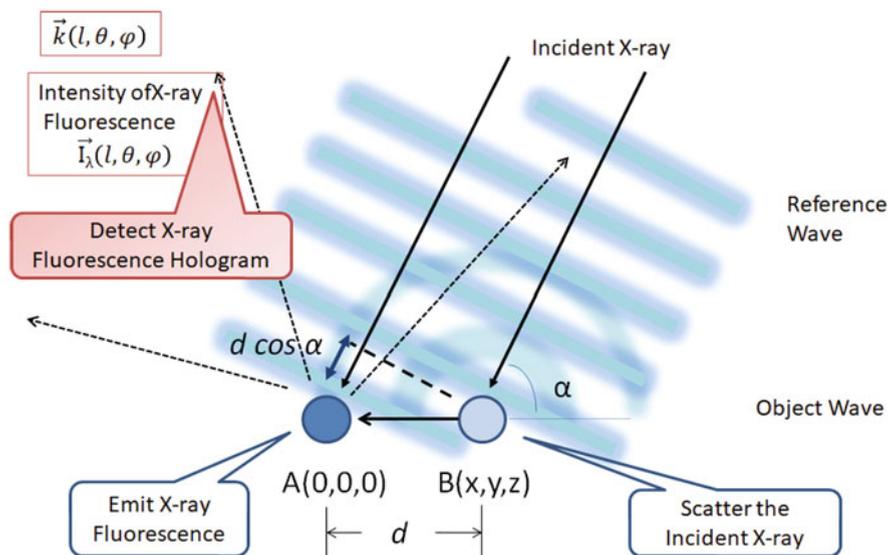


Fig. 2 Difference of optical path length in inverse mode

2.1 Reconstruction of Atomic Images

When the intensity $I_{\lambda}(l, \theta, \phi)$ measured at polar coordinate system $k(l, \theta, \phi)$ on a hologram on a sphere by incident X-ray of wave length λ , the position of an atom in the sample can be calculated as follows.

Suppose atom A and B are located at the origin and $\mathbf{r}(x, y, z)$ in the sample, the atomic image of atom b can be reconstructed by calculation similar to three-dimensional discrete Fourier Transform. In atomic images reconstructed from holograms obtained by several wave lengths, real atomic images are enhanced and, at the same time, ghost images are reduced[4]. In order to reduce the processing time, atomic images are reconstructed from the hologram on the sphere of radius l as shown in Eq. (1):

$$\chi(x, y, z) = \sum_{\theta} \sum_{\phi} \sum_{\lambda} -I_{\lambda}(\theta, \phi) \exp(i2\pi(|\mathbf{r}| - \mathbf{kr})/\lambda) \sin \theta \quad (1)$$

The input data are stored in double precision floating point number format. Coordinate on the sphere is represented in the polar coordinate system ranging from $\theta = 1^{\circ}$ to 179° and from $\phi = 0^{\circ}$ to 359° by 1° grid on each angle. The output data are grid points values on the rectangular coordinate system ranging from -10 to 10.0 \AA by 0.1 \AA grid on each axis. The complex numbers at grid points calculated by the reconstruction are stored in the output file.

192 grid points are laid on each axis on the rectangular coordinate system ranging from -9.6 to 9.6 \AA by 0.1 \AA grid in order to parallelize the reconstruction easily on PC cluster, it is explained in detail in Sect. 4.

Because the grid points of input data are located on the polar coordinate system and, on the other hand, those of output data are on the rectangular coordinate system, it is difficult to apply the fast Fourier transform (FFT) algorithm to DFT and it takes long time to calculate DFT.

We estimate that it may take a few days to reconstruct the three-dimensional atomic images from holograms measured by experiments. Because crystal structure of sample and its lattice constant are already known by other methods, in order to reduce the time required for reconstruct atomic images for crystal with a certain lattice constant, for example, 2 \AA , three-dimensional atomic images are estimated with several two-dimensional atomic images on x - y planes at $z = -4, -2, 0, 2, 4 \text{ \AA}$. If needed, atomic images at $z = -1.9$ and 2.1 \AA are reconstructed to analyze atomic fluctuations and lattice distortions.

Thus, it takes long time to analyze the crystal structure because reconstruction of two-dimensional atomic images and observation of the atomic images repeatedly.

2.2 Analysis Procedure of XFH

In XFH, experimental data are analyzed in the following procedure. The most time-consuming step is reconstruction while pre- and post- steps are also needed.

1. experiment
2. removal of background waves
3. completion of sphere data

4. reconstruction of atomic images and
5. display of them

Because it is known that low frequency noise called background wave is included in hologram data obtained by XFH experiments, the noise is removed before reconstruction.

In the experiment, the intensity of X-ray fluorescence emitted from sample is measured, while it is slanted from $\theta = 0$ to 75° and rotated from $\phi = 0$ to 360° at each θ in Fig. 1. For $\theta > 75^\circ$, the meaningful intensity is not measured because the surface of the sample and the incident X-ray are nearly parallel. By making use of symmetry of crystal structure of the sample, the hologram data on the sphere ranging from $\theta = 0$ to 180° are completed. An example of hologram on the sphere shown in Fig. 3 is obtained by their pre-processes such as removal of background waves and completion of sphere data.

The values on grid points of the reconstructed atomic images are merely values proportional to existence probability of atom at each grid point on the x-y-z rectangular coordinate system. For a grid point, the value of low existence probability is noise and its atom image is not displayed if the value is less than threshold value and reconstructed images are displayed as shown in Fig. 4.

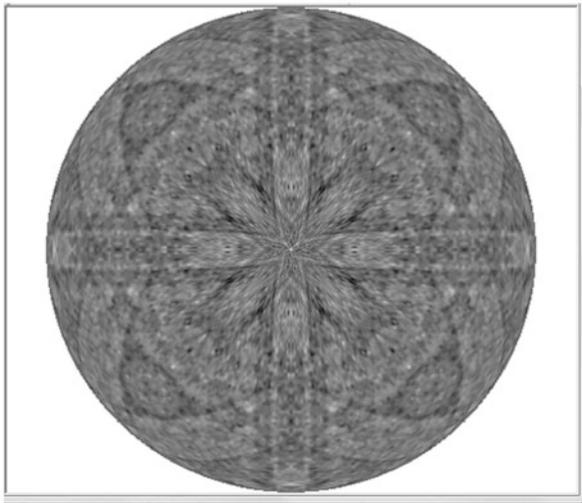
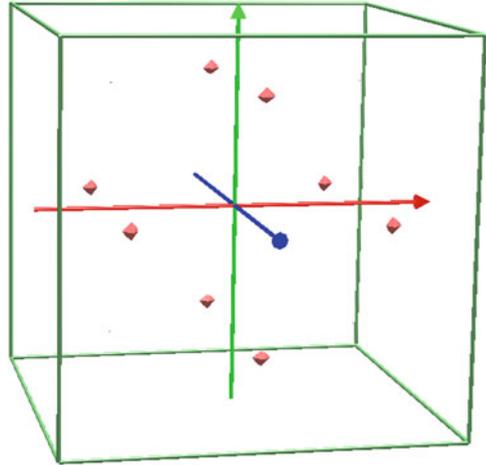


Fig. 3 An example hologram obtained by an experiment

Fig. 4 An example of reconstructed three-dimensional atom images



3 Parallelization

As pointed out in Sect. 2, it is an issue that it takes long time to analyze the crystal structure because reconstruction of two-dimensional atomic images and observation of the images are repeated several times. We therefore try to improve the reconstruction of three-dimensional atomic images by parallelizing DFT with parallel programming language XcalableMP and parallel API OpenMP on multi-node PC clusters.

OpenMP [7] is a parallel programming API which enables parallelization by inserting directives in sequential programs. It enables parallelization of loops with data parallelism easily and high performance on shared-memory machines can be achieved. Similar to OpenMP, XcalableMP [10] is a parallel programming language which also enables parallelization by inserting directives in sequential programs. Data distribution among distributed-memory computing environment such as multi-node PC clusters and supercomputers can be specified by inserting directives. Both OpenMP and XcalableMP are designed so that directives related to parallelization are ignored when the program is compiled to a sequential executable. Thus both sequential and parallel programs are maintained in one common source code. In addition, XcalableMP and OpenMP can be used at the same time for hybrid inter- and intra- node parallelization.

In this section, hybrid parallelization of reconstruction in two steps as follows:

1. parallelization of reconstruction of two-dimensional atomic images by OpenMP on single PC node and
2. parallelization of reconstruction of three-dimensional atomic images by XcalableMP and OpenMP on multi-nodes

3.1 Parallelization of Reconstruction of Two-Dimensional Atomic Images by OpenMP

Presently, according to the Eq. (1), in the loop of reconstruction of two-dimensional atomic images at certain z on x - y plane, five loops of x , y , θ , ϕ , and λ are nested from outer to inner loop. In this chapter, θ loop denotes the loop which accesses contiguous elements in a dimension of array θ for short.

The strategy of optimization and parallelization of two-dimensional reconstruction is as follows:

1. Replace trigonometric function calls with array references
2. Apply loop interchange and
3. Choose a loop to be parallelized

These optimization and parallelization are explained below.

In reconstruction of two-dimensional atomic images, some calls of trigonometric functions such as \sin and \cos in nested loops are replaced with references of arrays. The values of the trigonometric functions are stored in the arrays before entering the nested loops.

In this study, this nested loop is interchanged to λ , x , y , θ , and ϕ from outer to inner loop to improve the cache hit ratio. The size of the three-dimensional input array whose dimensions are λ , θ , and ϕ is about 10 MB and may not be stored on the secondary cache. In order to improve the cache hit ratio, when λ is fixed at the most outer loop, 500 KB data of the input array for a λ are stored on the cache and accessed repeatedly in the nested loop of x , y , θ , and ϕ . In the original loop nests, the index range of the inner-most λ loop is at most 20. After interchanging loops, the index range of the inner-most ϕ is 360 and it is expected that SIMD instructions in general purpose processors such as Intel AVX can be applied to the inner-most loop.

After these optimizations, the x loop in the nested loops is parallelized by OpenMP directive so that the performance of reconstruction of two-dimensional atomic images is improved.

3.2 Parallelization of Reconstruction of Three-dimensional Atomic Images by XscalableMP

In reconstruction of three-dimensional atomic images, according to the Eq. (1), by extending the optimized and parallelized nested loop of reconstruction of two-dimensional atomic images as described in Sect. 3.1, the six loops of λ , z , x , y , θ , and ϕ are nested from outer to inner loop. Similar to the two-dimensional loop, the values of some trigonometric functions are calculated in advance before entering the six nested loop.

Among the six nested loops, the outer z loop and the inner x loop are parallelized by XcalableMP and OpenMP. In other words, the nested loops are parallelized among inter-nodes by the z loop level and also parallelized within each node by the x loop level. The parallelized kernel loop of three-dimensional atomic image reconstruction is shown in Fig. 5. Other combinations of parallelizing x , y , and z loops are also performed and discussed in Sect 4.

In this study, z loop is parallelized by XcalableMP directives and the output arrays are distributed contiguously, namely BLOCK distribution, among nodes in z dimension as shown in Fig. 6. All of the input data are read on every node simultaneously and stored replicated. The data of atomic images, which are distributed among nodes in z dimension, are calculated on each node, aggregated to one root node as shown in Fig. 7, and finally written to the output file on the root node.

4 Performance Evaluation

In this section, we show the performance results of parallel runs of reconstruction of two-dimensional atomic images parallelized by OpenMP executed on a single node and reconstruction of three-dimensional atomic images parallelized by XcalableMP and OpenMP on a multi-node PC cluster. Comparison of XcalableMP and MPI for multi-node parallelization with respect to the performance and productivity of programming are also demonstrated.

The PC cluster consist of eight nodes and each node has two sockets of six-core Intel Xeon X5660 2.8 GHz and 24 GB main memory. The nodes are connected with InfiniBand DDR (4 Gbps) and Gigabit Ethernet. The size of smart cache on Xeon X5660 is 12 MB. The program is compiled with XcalableMP 1.2.2 and Intel Compiler 18.0.1 with `-O3 -XHOST` optimization option.

4.1 Performance Results of Reconstruction of Two-Dimensional Atomic Images

The program of reconstruction of two-dimensional atomic images is executed on a single node of the PC cluster. The input data are obtained in an experiment in which lead zirconate titanate (PZT) is used as a sample in the experiment and 21 types of incident X-rays are entered to the sample while varying angles ranging from $\theta = 1$ to 179° and from $\phi = 0$ to 359° . The output is a two-dimensional array $[x][y] = [192][192]$ for a certain z .

In Table 1, the execution time of the original, array reference of trigonometric function calls, and loop interchange on one node. Execution time and speed-up ratio

```

for (l=0; l<No_Incident; l++) {
    input_hologram(l, Xi);
#pragma xmp loop on t(z)
    for (iz=0; iz<NZ; iz++) {
        z = position(iz);
#pragma omp parallel for private(...)
        for (ix=0; ix<NX; ix++) {
            x = position(ix);
            for (iy=0; iy<NY; iy++) {
                y = position(iy);
                r_norm = norm(x,y,z);
                sum1r = 0.0;
                sum1i = 0.0;
                for (th=1; th<NTHETA; th++) {
                    sum2r = 0.0;
                    sum2i = 0.0;
                    for (phi=0; phi<NPHI; phi++) {
                        d = r_norm
                            - tab_kx[th][phi]*x
                            - tab_ky[th][phi]*y
                            - tab_kz[th]*z;
                        a = k[l]*d;
                        sum2r += Xi[th][phi]*cos(a);
                        sum2i += Xi[th][phi]*sin(a);
                    } /* end for phi */
                    sum1r += sum2r * tab_sin[th];
                    sum1i += sum2i * tab_sin[th];
                } /* end for theta */
                Ur_r[z][x][y] += sum1r * wt[l];
                Ur_i[z][x][y] += sum1i * wt[l];
            } /* end for y */
        } /* end for x */
    } /* end for z */
} /* end for l */

```

Fig. 5 Kernel loop of three-dimensional atomic image reconstruction

Fig. 6 Input and output arrays and specification of their distribution by XscalableMP directives

```
#pragma xmp nodes p(1)
#pragma xmp template t(0:191)
#pragma xmp distribute t(block) onto p

/* input array */
double Xi[NTHETA][NPHI];

/* output arrays */
double Ur_r[NZ][NX][NY];
#pragma xmp align Ur_r[n][*][*] with t(n)
double Ur_i[NZ][NX][NY];
#pragma xmp align Ur_i[n][*][*] with t(n)
double Ur_r1[NZ][NX][NY];
double Ur_i1[NZ][NX][NY];
```

Fig. 7 Aggregation of arrays by XscalableMP

```
#pragma xmp gmove
Ur_r1[:][:][:] = Ur_r[:][:][:];
#pragma xmp gmove
Ur_i1[:][:][:] = Ur_i[:][:][:];
```

Table 1 Performance results of reconstruction of two-dimensional atomic images by OpenMP on Xeon X5660

Original (s)	Array references and loop exchange (s)	OpenMP(s) 12 threads	Speed-up ratio
972.473	914.301	75.814	12.8

of the program parallelized by OpenMP with 12 threads on two sockets of six-core Xeon are 75.814 s and 12.8, respectively.

Here, let us consider the effect of the loop interchange. The size of input three-dimensional double precision array (λ, θ, ϕ) is about 10 MB. Before the loop interchange, loops x and y are the outer loops and 10 MB input data is repeatedly referenced in the nest of inner three loops λ , θ , and ϕ . Because the size of smart cache is 12 MB, it is assumed that the input data are spilled out of the cache and cache misses are occurred frequently. On the contrary, λ loop is placed at the outermost the loop nests by the loop interchange and a part of the input data is repeatedly referenced in the inner loops θ and ϕ . This fragment of the input data is about 500 KB and can be stored in the smart cache.

4.2 Performance Results of Reconstruction of Three-dimensional Atomic Images

The reconstruction of three-dimensional atomic images is executed mainly on the six nested loops of λ , z , x , y , θ , and ϕ .

The input and output data are stored in three-dimensional arrays of $[\lambda][\theta][\phi] = [21][179][360]$ and $[z][x][y] = [192][192][192]$, respectively. The output array is divided in z dimension and distributed among nodes on the PC cluster.

The z loop and x loop in the reconstruction program are parallelized by XcalableMP and OpenMP, respectively. The performance results of parallel execution on the PC cluster are shown in Tables 2 and 3. For the size of output $[z][x][y] = [192][192][192]$, because the estimation time of the sequential execution is too long, it is executed when the total number of threads is greater than or equal to eight as shown in Table 2. The #Threads columns of both Tables 2 and 3 stand for the total number of threads, which is the product of the number of nodes and the number threads per node. In addition to the total execution time, time for reading from the input file, aggregating the atomic images distributed among nodes, and writing to the output file.

The performance results of reconstruction of only for eight x - y planes at $Z = 0, 1 \dots 7$ are shown in Table 3. Because it takes a few hours in the sequential execution, this reconstruction is executed with threads ranging from 1 to 96.

The speed-up ratio of reconstruction of atomic images of both 192 and 8 x - y planes are depicted in Fig. 8. Both horizontal and vertical axes are logarithmic scales. The Z8 graph is the speed-up ratio to the one thread execution in Table 3. The Z192 graph is the speed-up ratio to the eight thread execution in Table 2 multiplied by 8. In both lines in Fig. 8, it is confirmed that nearly ideal speed-up is achieved. The speed-up ratio values at 96 threads for Z8 and Z192 are 94.21 and 94.23, respectively.

Table 2 Performance results for reconstruction (z:192) parallelized by XcalableMP

#Threads	Execution (s)	Input (s)	Aggregation (s)	Output (s)
8 (8 × 1)	21,683.076	0.781	0.176	9.772
48 (8 × 6)	3,623.352	0.721	0.163	9.731
96 (8 × 12)	1,840.942	0.767	0.174	9.701

Table 3 Performance results for reconstruction (z:8) parallelized by XcalableMP

#Threads	Execution (s)	Input (s)	Aggregation (s)	Output (s)
1 (1 × 1)	7,214.325	0.745	0.007	0.301
8 (8 × 1)	923.830	0.473	0.009	0.303
12 (1 × 12)	603.962	0.459	0.005	0.310
48 (8 × 6)	151.574	0.458	0.008	0.303
96 (8 × 12)	76.576	0.455	0.009	0.310

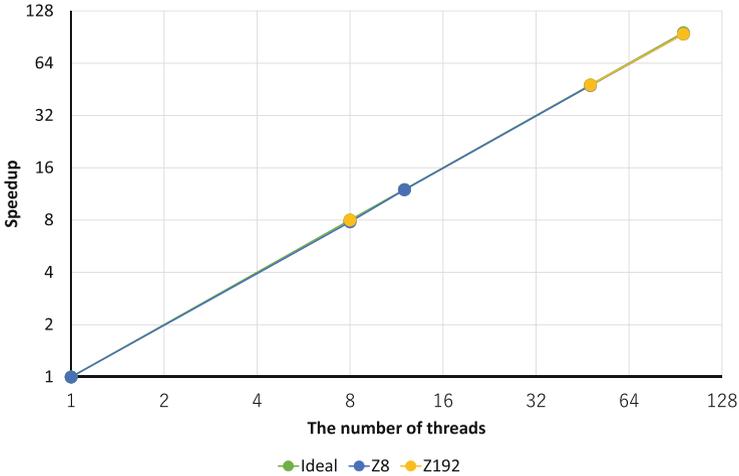


Fig. 8 The speed-up ratio of execution parallelized by XcalableMP

Because the sizes of input and output data are small, the ratio of the file I/O time to the total execution time is very small and the time of aggregation of distributed data to one root node by gmove construct of XcalableMP is also very short. Therefore, high performance is achieved without parallel file I/O.

4.3 Comparison of Parallelization with MPI

In order to compare the productivity of parallel programming, we also implemented the reconstruction of three-dimensional atomic images with MPI. The number of lines of the program in C already parallelized by OpenMP is 350 and the number of modified or inserted lines for multi-node parallelization by XcalableMP is 32, while that by MPI is 53. This program can be parallelized with less effort in XcalableMP than in MPI.

Table 4 summarizes the execution time and the number of modified and inserted lines. The size of the reconstructed atomic images is $[z][x][y] = [192][192][192]$ and 96 threads are used in total on the eight-node PC cluster. The difference of the execution time parallelized by XcalableMP and MPI is small. We confirmed that the higher productivity of parallel programming is achieved by XcalableMP than MPI without sacrificing performance.

Table 4 Execution parallelized by XcalableMP and MPI (z:192, 96 threads)

Parallelization	Time (s)	#Modified lines
XcalableMP	1, 840.942	32
MPI	1, 817.042	53

Data aggregation of the reconstructed atomic images distributed among nodes is described by `gmove` statement by XcalableMP, while called `MPI_Gatherv` library function by MPI. We confirmed XcalableMP compiler transfers `gmove` to appropriate MPI library functions.

5 Conclusion

This chapter describes parallelization of reconstruction of three-dimensional atomic images in X-ray fluorescence holography, which is an analysis method of material science. In order to execute it on large-scale PC clusters and supercomputer, we adopt hybrid parallelization, or inter- and intra-node parallelization by XcalableMP and OpenMP.

The program, whose input is 21 holograms data of 179×360 points and output is a three-dimensional atomic image of 192^3 points, is executed on PC cluster which consists of eight nodes of Intel Xeon X5660 processors and 96 cores in total, is executed in 1841 s, or about half an hour. We estimated that it would take a few days to execute this reconstruction sequentially. We confirmed that the performance is improved by parallelization to the practical use.

We also confirmed that the higher productivity of parallel programming is achieved by XcalableMP than MPI without sacrificing performance.

Acknowledgments The part of this work was supported by JSPS Grant-in-Aid for Scientific Research on Innovative Areas “3D Active-Site Science,” Grant Number 26105013.

References

1. J.J. Barton, Photoelectron holography. *Phys. Rev. Lett.* **61**(12), 1356–1359 (1988)
2. J.J. Barton, Removing multiple scattering and twin images from holographic images. *Phys. Rev. Lett.* **67**(22), 3106–3109 (1991)
3. B.L. Chamberlain, Chapel, chapter 6, in *Programming Models for Parallel Computing*, ed. by P. Balaji (The MIT Press, Cambridge, 2015)
4. K. Hayashi, N. Happo, S. Hosokawa, Evaluation of local lattice distortion by X-ray fluorescence holography. *JSSRR* **26**(4), 195–205 (2013) [in Japanese]
5. High Performance Fortran Forum, *High Performance Fortran Language Specification (Ver. 2.0)* (1997)
6. Message Passing Interface Forum, *MPI: Message Passing Interface Version 3.1* (2015), <https://www.mpi-forum.org/docs/mpi-3.1/mpi31-report.pdf>
7. OpenMP Architecture Review Board, *OpenMP Application Program Interface Version 5.0* (2018), <https://www.openmp.org/wp-content/uploads/OpenMP-API-Specification-5.0.pdf>

8. V. Saraswat, B. Bloom, I. Peshansky, O. Tardieu, D. Grove, *X10 Language Specification Version 2.6.2* (2019), <http://x10.sourceforge.net/documentation/languagespec/x10-latest.pdf>
9. UPC Consortium, *Berkeley UPC – Unified Parallel C Version 2019.4.2* (2019), <http://upc.lbl.gov>
10. XcalableMP Specification Working Group, *XcalableMP Language Specification Version 1.4* (2018), <https://xcalablemp.org/download/spec/xmp-spec-1.4.pdf>

Open Access This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.

