数理解析研究所講究録
第 1995 巻 2016 年 102-111

Communication-Avoiding CG Method: New Direction of Krylov Subspace Methods towards Exa-scale Computing

Reiji SUDA[1], Cong LI[1], Daichi WATANABE[1],
Yosuke KUMAGAI[2], Akihiro FUJII[2], Teruo TANAKA[2]
[1] the University of Tokyo, [2] Kogakuin University

1 Introduction

Current supercomputers, such as K computer and Chinese Tianhe 2, have close to, or even more than one million cores. The supercomputers in the next generation will have still higher degrees of parallelism. It is expected that supercomputers will have performance of 10^{18} or 1Exa flops, and called *exa-scale computers*. They would consist of about one million nodes, and the parallelism within each node will be about one thousand, with combined multicore and SIMD parallelisms.

Several challenges are anticipated in effective programming on such an extremely large scale parallel computers. Some eminent challenges must be power consumption, programming complexity, fault tolerance, memory barrier, and communication latency. In this article, we focus on the problem of higher communication latency in exa-scale computers. The problem of communication latency is not new. In the past, the relative cost of communication latency was kept under a certain limit by assigning large enough computations to all processors, so that the processors are busy enough. This idea leads to *weak scaling*, in which the total size of the computation is increased linearly to the number of processors. However, now the number of processors is so big, and thus the problem size is extremely big. Users may not want to make the problem still larger, but may want to reduce the computation time by keeping the problem size and using more processors. This idea of *strong scaling* must be carefully introduced, because communication latency could bring serious performance degradations: The computations are distributed into more processors, and the amount of computations per processor will become less, while more messages might be required for the larger number of processors to communicate each other. The resulting effect brings a situation that larger part of execution time is occupied by communication. Even now on K computer, some application program spends much longer time for communication than time for computation.

Those observations leads people into research on a class of algorithms in which the amount of communication or the number of messages are reduced at the cost of increased amount of computations, compared to existing algorithms. Such algorithms are collectively called *Communication-Avoiding* or *CA* algorithms. Communication-avoiding algorithms may reduce the total execution time of some parallel computations, especially of strong scaling on extremely large scale parallel computers.

Perhaps the most famous and successful communication-avoiding algorithm in dense matrix computations is TSQR (Tall-Skinny QR) algorithm[1]. Various kinds of temporal blocking algorithms[2, 3] have been proposed for stencil computations. Communication-avoiding iterative sparse solvers, which is the topic of this article, are also investigated with great efforts[4]. We consider algorithms that reduces the number of messages (on each processor) in this article.

2 Communication-Avoiding Krylov Subspace Methods

2.1 Conjugate Gradient (CG) Method and its Communication

The following pseudo-code shows the Conjugate Gradient (CG) method (without preprocessing), which is one of the most basic Krylov subspace methods of iterative linear solvers. Here, (u, v) in the expression represents an inner product of vectors u and v, and the suffixes are used to identify iteration counts.

```
Inputs: $A$, $b$, $x_0$
$p_0 = r_0 = b - Ax_0$
for $i = 0, 1, 2, \dots$
    $\alpha_i = (r_i, r_i)/(p_i, Ap_i)$
    $r_{i+1} = r_i - \alpha_i Ap_i$
    $x_{i+1} = x_i + \alpha_i p_i$
    $\beta_i = (r_{i+1}, r_{i+1})/(r_i, r_i)$
    $p_{i+1} = r_{i+1} + \beta p_i$
```

In parallelizing CG method, communication is needed to compute one matrix-vector product Ap_i and two inner products (r_i, r_i) and (p_i, Ap_i). The other computations are scalar-vector products, vector additions/subtractions, and scalar operations, which requires no communication. In the following, we first focus on the communication for the inner products.

2.2 Reduction of the Number of Inner Products in the CG Method

It is known that the number of inner products in the CG method can be reduced from two to one (Saad[5], Meurant[6], D'Azevedo[7]). Such a reduction of the number of inner products becomes possible by introducing a matrix-matrix multiplication of very thin matrices. There are some methods with slight differences as cited above. They will be effective on some large scale parallel computers, perhaps are effective on some contemporary computers already. Their convergence might be affected by rounding errors incurred by the different expressions employed for the algorithm. We do not discuss them more in this article.

In order to further reduce the number of inner products, one way is to choose a positive integer k and to collect the computations of k steps of i loop into one, which is a well-known technique as (k-way) look unrolling. This is known as multi-step CG method. One of oldest work in this kind is one by van Rosendale[8]. Chronopoulos and Gear[9] proposed s-step method which is multi-step CG method. Chronopoulos and Kucherov[10] discusses blocking of s-step method. Thesis by Toledo[11] proposed new multi-step methods. Thesis by Hoemmen[4] is convenient to survey recent works.

Krylov subspace methods, including the CG method, can construct approximate solutions if the Krylov subspace:

$$K_m = \mathrm{span}\{r_0, Ar_0, A^2r_0, \dots, A^m r_0\}$$

is available. Applying this technique to the CG method, it leads to an iterative solver with the number of inner products is $1/k$ times those of the CG method.

But the above straightforward construction of Krylov subspace, which is known as *monomial basis*, is not numerically stable. This is easily understood when one notices that the computation of $A^m r_0$ is mathematically equivalent to the power method. The vector $A^m r_0$ is almost parallel to the eigenvector of the eigenvalue with largest absolute magnitude, and the other eigen-components decrease relatively to the largest one and hidden by the rounding errors. As a result, numerical dimension of the Krylov subspace becomes shorter than expected, and the approximate solution does not converge to the analytical solution well. In experiments, a slowdown of the convergence is observed if the unrolling factor k is larger than 2 to 5, in many cases.

To avoid such degradation of convergence, we use Chebyshev polynomial $T_m(x)$ to generate Krylov subspace as:

$$S = (r, Ar, T_2(A)r, \dots, T_m(A)r).$$

Here, $T_m(x)$ represents an orthogonal polynomial on the interval $[\lambda_{\min}, \lambda_{\max}]$ (where $\lambda_{\min}$ and $\lambda_{\max}$ are the smallest and the largest eigenvalues of A) by scaling and shifting the usual Chebyshev polynomial on the interval $[-1, 1]$. Chebyshev polynomial $T_m(x)$ has all of its local maxima, local minima and zeros in the interval $[\lambda_{\min}, \lambda_{\max}]$, and the magnitudes of the local maxima and local minima are all 1's. Zeros of Chebyshev polynomials of different degrees never coincide with each other. And exactly one of the zeros of $T_m(x)$ exists in between two neighboring zeros of $T_{m-1}(x)$. Those properties of the Chebyshev polynomial guarantee that no one eigenvector of A will not surpass others in the Krylov subspace, and numerically stabilize the generation of the Krylov subspace. The following is our formulation of such an algorithm:

```
Inputs: $A$, $b$, $x_0$, $M^{-1}$
$r_0 = b - Ax_0$
for $i = 1, 2, 3, \dots$
    $r = r_{(i-1)k}$
    $S_i = M^{-1}(r, AM^{-1}r, T_2(AM^{-1})r, \dots, T_{k-1}(AM^{-1})r)$
    if $i > 1$ then
        $B_{i-1} = (Q_{i-1}^{\top} A Q_{i-1})^{-1} Q_{i-1}^{\top} A S_i$
        $Q_i = S_i - Q_{i-1} B_{i-1}$
    else
        $Q_i = S_i$
    $a_i = (Q_i^{\top} A Q_i)^{-1} Q_i^{\top} r$
    $x_{ik} = x_{(i-1)k} + Q_i a_i$
    $r_{ik} = r_{(i-1)k} - A Q_i a_i$
```

which we call *Chebyshev-Basis CG* or *CBCG* method[12]. Derivation will be given in the next subsection. But in the above pseudo-code we include the preconditioning $M^{-1} \approx A^{-1}$.

The above pseudo-code only provides its mathematical formulation. A straightforward implementation of it would contain much more matrix-vector products than the original CG method, but, as Kumagai et al.[14] pointed out, it can be implemented with just k matrix-vector products. We do not apply elaborate preconditioning more than Jacobi preconditioner, but there are proposals of preconditioners suitable to communication-avoiding Krylov subspace method, for example Yamazaki et al.[15] proposed a strong preconditioner without introducing additional communication.

2.3 Review of the CG Method

The CG method solves a linear equation

$$Ax = b,$$

where the coefficient matrix A must be symmetric positive definite. There are several derivations and several implementation details of CG method, and the followings are those on which our CBCG method is based on.

The CG method starts with an initial guess of the solution x_0 (which can be just 0), and constructs approximation solutions x_1, x_2, $\dots$, one for each iteration. The approximate solution x_i is constructed as

$$x_i = x_0 + \sum_{j=0}^{i-1} \alpha_j p_j,$$

where p_j's are vectors called search direction vectors. Thus, the approximate solution is the sum of initial guess x_0 and a linear combination of search direction vectors p_j.

Search direction vectors must be bi-orthogonal:

$$p_i^{\top} A p_j = 0 \quad (i \neq j).$$

First we proceed the derivation of the CG algorithm assuming availabilities of p_j's, and after that we will explain how to obtain such vectors.

The coefficients α_i's are chosen so that the A-norm of the error

$$(x - x_i)^{\top} A (x - x_i) = (x - x_0 - \sum_{j=0}^{i-1} \alpha_j p_j)^{\top} A (x - x_0 - \sum_{j=0}^{i-1} \alpha_j p_j)$$

is minimized. The A-norm of the error can be represented as

$$(x - x_i)^{\top} A (x - x_i) = e_0^{\top} A e_0 - 2 e_0^{\top} A \sum_{j=0}^{i-1} \alpha_j p_j + \sum_{j=0}^{i-1} \alpha_j^2 p_j^{\top} A p_j,$$

where $e_0 = x - x_0$. Differentiating it by α_j and then equating them to zeros, we have:

$$\alpha_j = \frac{e_0^{\top} A p_j}{p_j^{\top} A p_j}.$$

Thus the coefficients α_j's are determined.

Next, the construction of search direction vectors p_j's is explained. They are created by Gramm-Schmidt A-orthogonalization of Krylov subspace

$$K_i = \mathrm{span}\{r_0, Ar_0, A^2 r_0, \ldots, A^i r_0\},$$

which results in:

$$p_i = r_i - \sum_{j=0}^{i-1} \frac{p_j^{\top} A r_i}{p_j^{\top} A p_j} p_j. \tag{1}$$

Here a property called Lanczos principle gives an important equality

$$p_j^{\top} A r_i = 0 \quad (j < i - 2)$$

(which we omit the proof), and they simplify the equation (1) of orthogonalization as

$$p_i = r_i + \beta_{i-1} p_{i-1}, \qquad \beta_{i-1} = -\frac{p_{i-1}^{\top} A r_i}{p_{i-1}^{\top} A p_{i-1}}.$$

By summarizing the above results, the CG method is defined as in Section 2.1. There, some expressions different from those shown in the derivation are employed: some reduces rounding errors, and some reduces computational costs.

2.4 Deriving the CBCG Method

It is known that the Krylov subspace has several useful properties such as:

$$\begin{aligned} K_i &= \mathrm{span}\{r_0, Ar_0, A^2 r_0, \ldots, A^i r_0\} \\ &= \mathrm{span}\{r_0, r_1, r_2, \ldots, r_{i+1}\} \\ &= \mathrm{span}\{p_0, p_1, p_2, \ldots, p_{i+1}\}. \end{aligned}$$

As it is clear from the above explanation of the CG method, it holds that

$$x_i - x_0 = \sum_{j=0}^{i-1} \alpha_j p_j \in K_i,$$

which means that the approximate solution x_i is chosen from the Krylov subspace so to minimize A-norm of the error.

Our strategy is to create the same Krylov subspace as the CG method (except rounding errors). In the following presentation, i is used for the index of outer (unrolled) loop, and j is used for the index of inner loop which runs from 0 to $k-1$. Then $ik+j$ corresponds to the loop index of the original CG method.

In the CBCG method, k dimensions of Krylov subspace are created at once. Let $q_{i,j}$ be the vectors that spans the Krylov subspace:

$$K_{ik} = \text{span}\{q_{1,1}, q_{1,2}, \ldots, q_{1,k}, q_{2,1}, q_{2,2}, \ldots, q_{i,k}\}.$$

Also let Q_i be a matrix which consists of k vectors of $q_{i,j}$:

$$Q_i = (q_{i,1}, q_{i,2}, \ldots, q_{i,k}).$$

Next we introduce a constraint on Q_i (this constraint is not unique, and other choices can be considered):

$$Q_{i_1}^\top A Q_{i_2} = 0 \qquad (i_1 \neq i_2),$$

meaning that the Q_i matrices are A-orthogonal. But vectors within one Q_i matrix may not be A-orthogonal (unlike CG method). Thus we may have

$$q_{ij_1}^\top A q_{ij_2} \neq 0.$$

As the approximate solution is contained in the Krylov subspace, it has the form as:

$$x_{ik} = x_0 + \sum_{i,j} \alpha_{i,j} q_{i,j}.$$

It can be written as

$$x_{ik} = x_0 + \sum_{\iota=0}^{i-1} Q_\iota a_\iota$$

by using a vector consisting of $\alpha_{i,j}$

$$a_i = (\alpha_{i,1}, \alpha_{i,2}, \ldots, \alpha_{i,k})^\top.$$

First, the coefficient a_i is determined from the A-norm of the error

$$\begin{aligned}
&(x - x_{ik})^\top A (x - x_{ik}) \\
&= (x - x_0 - \sum_{\iota=0}^{i-1} Q_\iota a_\iota)^\top A (x - x_0 - \sum_{\iota=0}^{i-1} Q_\iota a_\iota) \\
&= e_0^\top A e_0 - 2 e_0^\top A \sum_{\iota=0}^{i-1} Q_\iota a_\iota + \sum_{\iota=0}^{i-1} a_\iota^\top Q_\iota^\top A Q_\iota a_\iota,
\end{aligned}$$

where $e_0 = x - x_0$ again. Differentiating it by a_i and equating it to zero, we have

$$a_i = (Q_i^\top A Q_i)^{-1} Q_i^\top r_0.$$

Next construction of $q_{\iota,j}$ is explained. In the original CG method, the search direction vector p_i is created by Gramm-Schmidt A-orthogonalization from the residual vector r_i. In CBCG method, the search direction vector $q_{i,j}$ is again constructed by Gramm-Schmidt A-orthogonalization from some vector $s_{i,j}$. That is, by letting

$$S_i = (s_{i,1}, s_{i,2}, \dots, s_{i,k}),$$

Gramm-Schmidt reads as

$$Q_i = S_i - \sum_{\iota=0}^{i-1} Q_\iota (Q_\iota^\top A Q_\iota)^{-1} Q_\iota^\top A S_i.$$

Similar to the CG method, most of the terms in the above formula are not required. Let

$$B_{i-1} = -(Q_{i-1}^\top A Q_{i-1})^{-1} Q_{i-1}^\top A S_i,$$

and then it is similar to β_i in the CG method. We have

$$Q_\iota = S_i + Q_{i-1} B_{i-1}, \tag{2}$$

as we will prove it later.

To expand the same Krylov subspace as the CG method, $s_{i,j}$ must be in $K_{ik+j} - K_{ik+j-1}$. For brevity, we temporary let $r = r_{(i-1)k}$. Then from the fact that $r \in K_{(i-1)k+1}$, it can be seen that

$$s_{i,j} = A^{j-1} r$$

satisfies $s_{i,j} \in K_{ik+j}$ (this is the monomial basis).

Finally we show equation (2). In the original CG method, it is known that

$$\forall a \in K_{i-1}, \quad r_i^\top v = 0$$

and

$$A^j r_i \in K_{i+j}$$

are satisfied. First assume that $\iota + j < i$. Then $A^j r_\iota \in K_{\iota+j}$ implies that $A^j r_\iota$ and r_i are orthogonal. Then the right hand side of

$$(A^j r_i)^\top r_\iota = r_i^\top (A^j r_\iota)$$

must be 0, and $A^j r_\iota$ and r_ι are orthogonal. From $r = r_{(i-1)k}$ and

$$s_{i,k} \in \operatorname{span}\{r, Ar, A^2 r, \dots, A^{k-1} r\},$$

it is shown that all of $As_{i,1}, As_{i,2}, \dots, As_{i,k}$ are orthogonal to all vectors from r_0 to $r_{(i-2)k}$. Thus AS_i is orthogonal to any vector in $K_{(i-2)k+1}$, and thus it is also orthogonal to all column vectors found in Q_0 to Q_{i-2}. Thus Gramm-Schmidt A-orthogonalization is simplified as equation (2).

As is seen in the derivation (and also in the pseudo-code), our CBCG method is a straightforward extension of the original CG method. There are two inner-product-like communication, as is shown by Kumagai et al.[14]. It could be combined to one, but we prioritize the numerical stability. However, the stability of the CBCG method is not perfect. One reason is that the Chebyshev polynomial is not perfect for all matrices. Another point of instability is at the inverse of $Q_{i-1} A Q_{i-1}$, which is not necessarily non-singular.

2.5 Numerical Instability of Communication-Avoiding CG Methods

Chebyshev polynomial is not perfect. For some matrices, k can be chosen almost arbitrarily without affecting convergence, but for some other matrices, a larger k degrades the convergence because of rounding errors. We have an unpublished work of generating Krylov subspace in a numerically more stable manner than the CBCG method, which will be reported in future.

In this paper, we review our previous work[16], where we have investigated the influence of the rounding errors and the corruption of the Krylov subspace into convergence, by using monomial basis in a communication-avoiding CG method presented in Hoemmen's work[4], modified in an introduction of restarts based on Beale's method[17].

We had executed numerical experiments using 25 matrices from the University of Florida Sparse Matrix Collections [18]. The matrices are small ones, from 27×27 "ex5" matrix to 2910×2910 "nasa2910" matrix. Jacobi preconditioner is applied, and the convergence of A-norm of the residual vector is observed (but the stopping criteria is based on 2-norm of the residual).

The convergence of static iterative methods, such as Jacobi, Gauss-Seidel and SOR methods, is asymptotically *linear* for symmetric matrices: that is, the norm of the residual vector decreases by a constant factor (which is between 0 and 1) by one iteration. The convergence of Krylov subspace methods, including the CG method, can be *superlinear*, that is, the residual norm decreases faster than linear. In our observations, we classify convergences of the CG method into three types: The first type is *linear*, where the observed convergence till the stopping criteria is almost linear. The numbers of iteration counts to the stopping criteria tend to be relatively large, and thus this *linear* convergence implies a slow convergence. The second type is *accelerated*, where the convergence of the residual becomes faster as iterations proceed. This is typical superlinear convergence. The third type is *stairs-like* convergence, where the convergence is faster intermittently, after a significant number of iterations with almost stagnated residual norms. Such convergence is also frequently observed in the CG method.

Next we observed the convergence of communication-avoiding CG method with different values of k. The major criterion that we chose for comparison is the effective numbers of inner products, defined by the number of outer iterations (indexed by i in the previous pseudo-code) divided by k. The number of outer iterations includes iterations lost by restart, since our restart rolls back the iterations when it detects divergence. For many problems of linear type convergence, the convergence was insensitive to the choice of k value. In such cases, choosing large k gives small numbers of inner products. For many problems of superlinear (accelerated type and stairs-like type) convergence, a large value of k degrades the convergence. The superlinear convergence ceases to appear, and the number of inner products increases because many iterations are required until convergence. In such cases, the value of k should be carefully chosen so that the convergence is not affected much.

One can expect that the best choice of k is related to the condition number (when the monomial basis is applied). We compared the "optimal" k that gives smallest number of inner products until convergence, against the $\hat{k}$ that satisfies $\kappa^{\hat{k}} \approx 10^{16}$, where κ is the condition number of the coefficient matrix, and 10^{16} is the inverse of machine epsilon. In many cases the optimal k is larger than $\hat{k}$, and significantly larger in cases of linear convergence. In cases of accelerated convergence, k can be more than twice as $\hat{k}$, and in cases of stairs-like convergence, the optimal k is close to $\hat{k}$. We also noticed that stairs-like convergence is observed with matrices with relatively large condition numbers ($\geq 10^6$).

2.6 Matrix Powers Kernel

As stated above, in the generation of Krylov subspace S_i, we need k matrix-vector products. The communication in those k matrix-vector products is exactly same for the Chebyshev basis and the monomial basis, as three-term recurrence formula of the Chebyshev polynomial can be used. So it is enough to investigate *Matrix Powers Kernel* or *MPK*:

$$(r, Ar, A^2r, \dots, A^{k-1}r).$$

If the linear equation (i.e. the coefficient matrix A) is derived by discretizing a partial differential equation (such as Poisson's equation) with regular mesh, the communication also coincides with stencil computations. In such cases, the vast existing knowledge about temporal blocking strategies of stencil computations can be directly utilized in communication-avoiding Krylov subspace methods.

However, in such cases where the linear equation is derived by FEM (Finite Element Method) with irregular mesh, one has to assume quite a general structure for the sparsity pattern of matrix A, usually distributed after partition by a general purpose graph partitioner. Research on efficient and effective Matrix Powers Kernel constructions for such general sparse matrices is still in an early stage. Basic methods are those called PA1 and PA2 in the paper by Hoemmen[4]. They require just one communication for k iterations, but a drawback is duplicated computations, that is, multiple processors do the same computations independently (PA2 has less duplicated computations than PA1). That incurs further increase of computational cost, and sometimes the increase of computational cost is higher than the communication cost reduction. Our resent work on Matrix Powers Kernel is presented in a paper[19]. There we show an algorithm that removes duplicated computations at the cost of larger number of communication operations.

2.7 Implementation of CBCG Method on K Computer

In recent paper[14], we have implemented the CBCG method on K computer, which is the largest supercomputer in Japan at the time of writing this article. The matrix of size $300^3 = 27,000,000$ is defined by discretization of a Poisson equation which appears in a computational fluid dynamics analysis. The values for k are 10 and 20.

The numbers of iterations until convergence were almost same as that of the CG method, and so the convergence was not degraded by rounding errors. When we used 12,288 or less numbers of nodes, the original CG implementation was faster, but when we use more 24,576 or larger numbers of nodes, the CBCG implementation became faster than the CG method. With larger numbers of nodes, a larger k tends to be more effective.

3 Conclusion

In this article, we have introduced basic concepts of communication-avoiding Krylov subspace methods, and summarizes our efforts on them. At reduction of the number of inner products, the numerical stability is slightly degraded, and iteration counts to convergence increase. The instability is much reduced by using Chebyshev basis, but the current technology is not fully satisfactory.

One of most serious questions arises from the existence of matrix inverse as $(Q_{i-1}AQ_{i-1})^{-1}$. It is not guaranteed that $Q_{i-1}AQ_{i-1}$ is full rank. We do not know what is the optimal way to cope with the situation that $Q_{i-1}AQ_{i-1}$ is singular or numerically singular. There are many other questions to answer: How should we choose the value of k for a given problem? What is the most efficient way to estimate $\lambda_{\min}$ and $\lambda_{\max}$? Which algorithm among many CA-CG methods is best for given platform and given problem? Do we have a better basis than the Chebyshev basis, if more information about eigenvalues is available? For Matrix Powers Kernel, we have much more things to investigate. There are many temporal blocking strategies in stencil computations, but a few are extended to general matrix. Graph partitioners may need to be redesigned toward Matrix Powers Kernel. Effective and efficient implementations should be investigated. We still need to spend more efforts toward answers to those questions.

Acknowledgements Part of this research is supported by JST CREST "An Evolutionary Approach to Construction of a Software Development Environment for Massively-Parallel Computing Systems", JSPS KAKANHI 15H02708 and 15K12033.

References

[1] J. Demmel, L. Grigori, M. Hoemmen, J. Langou, Communication-optimal Parallel and Sequential QR and LU Factorizations, SIAM J. Sci. Comput. 34(1), pp. A206–A239, 2012.

[2] T. Muranushi and J. Makino, Optimal Temporal Blocking for Stencil Computation, Procedia Computer Science, Vol. 51, 2015, pp. 1303-1312 (Proc. ICCS 2015).

[3] D. G. Wonnacott, M. M. Strout, On the Scalability of Loop Tiling Techniques, Proceedings of IMPACT 2013, Berlin, 2013, pp. 3-11.

[4] M. Hoemmen, Communication-Avoiding Krylov Subspace Methods, Ph.D thesis, UC Berkeley, 2010.

[5] Saad, Y.: Krylov subspace methods on supercomputers, *SIAM J. Sci. Statist. Comput.*, Vol.10, pp.1200-1232, 1989.

[6] Meurant, G.: Multitasking the conjugate gradient method on the CRAY X-MP/48, *Parallel Computing*, Vol.5, pp.267-280, 1987.

[7] D'Azevedo, E. F., Eijkhout, V. L., and Romine, C. H.: Conjugate gradient algorithms with reduced synchronization overhead on distributed memory multiprocessors, *LAPACK working note 56*, 1999.

[8] van Rosendale, J.: Minimizing inner product data dependencies in conjugate gradient iteration, NASA contractor report, 1983.

[9] Chronopoulos, A. T. and Gear, C. W.: s-step iterative methods for symmetric linear systems, *Journal of Computational Applied Mathematics*, Vol.25, pp.153-168, 1989.

[10] Chronopoulos, A. T. and Kucherov, A. B.: Block s-step Krylov iterative methods, *Numerical Linear Algebra with Applications*, Vol.17, pp.3-15, 2010.

[11] Toledo, S. A.: Quantitative Performance Modeling of Scientific Computations and Creating Locality in Numerical Algorithms, *Ph.D. thesis, EECS, MIT*, 1995.

[12] R. Suda and T. Motoya, Chebyshev-Basis Conjugate Gradient Method, IPSJ HPCS 2013, poster presentation. (In Japanese)

[13] R. Suda, C. Li and K. Shimane, Numerically Stable Communication-Avoiding Krylov Subspace Method, Proc. 19th JSCES Annual Meeting, 2014. (In Japanese)

[14] Y. Kumagai, A. Fujii, T. Tanaka, Y. Hirota, T. Fukaya, T. Imamura and R. Suda, Performance Analysis of Chebyshev Basis Conjugate Gradient Method on the K Computer, Proc. 11th International Conference on Parallel Processing and Applied Mathematics (PPAM2015).

[15] I. Yamazaki, S. Rajamanickam, E. Boman, M. Hoemmen, M. Heroux, and S. Tomov: Domain decomposition preconditioners for communication-avoiding Krylov methods on a hybrid CPU/GPU cluster. Proc. SC 2014: 933-944

[16] D. Watanabe, R. Suda, Choice of Extension Number of Basis Vectors in Communication-Avoiding CG Method, Proc. 20th JSCES Annual Meeting, 2015. (In Japanese)

[17] M. J. D. Powell: Restart procedure for the conjugate gradient method, Mathematical programming, Vol, 12(1), pp. 241-254, 1977.

[18] T. A. Davis and Y. Hu, The University of Florida Sparse Matrix Collection.

[19] R. Suda, Toward Generalized Matrix Powers Kernel, Proc. JSIAM Annual Meeting, 2015. (In Japanese)