Proceedings of the World Congress on Engineering 2015 Vol I
WCE 2015, July 1 - 3, 2015, London, U.K.

An Estimation of Single-Synchronized Krylov Subspace Methods with Hybrid Parallelization

Seiji Fujino, Kosuke Iwasato

***Abstract*—We evaluate performance of parallel computing of revised BiCGSafe and BiCGStar-plus method. Through several numerical experiments, we will make clear that the revised single synchronized BiCGSafe method outperforms other methods from the view points of elapsed time and speed-up on parallel computer with distributed memory.**

***Index Terms*—Krylov subspace method, Synchronization, Parallelization**

I. INTRODUCTION

We consider iterative methods for solving a linear system of equations $A\boldsymbol{x} = \boldsymbol{b}$ where $A \in R^{N\times N}$ is a given non-symmetric matrix. Vectors $\boldsymbol{x}$ and $\boldsymbol{b}$ are a solution vector and a right-hand side vector, respectively. Among many iterative methods, product-type of iterative methods, e.g., BiCGStab (Bi-Conjugate Gradient Stabilized)[6] and GPBiCG[7] are often used for the purpose of solution for realistic problems. However, the number of synchronizations per one iteration needs three times. BiCGSafe (with safety convergence) method[2] using the strategy of associate residual was proposed in 2005. This strategy leads to reduce the number of synchronization from three to two times per one iteration. BiCGStar (with stabilization of associate residual) method using the this stratgy and three-term recurrence as stabilized polynomial was proposed in 2013. BiCGStar[3] has two synchronization points per one iteration. The variants of GPBiCG method[1] improved GPBiCG itself by using the three-term recurrence similar to the one for the Lanczos polynomials. These variants of GPBiCG method needs two times synchronization per one iteration. We adopted the above formula for computation of parameters α_k and β_k to reduce the number of synchronization of from two times to single time per one iteration.

In this paper, we evaluate performance of parallel computing of revised BiCGSafe and BiCGStar-plus method. Through several numerical experiments, we make clear that the revised single synchronized BiCGSafe method outperforms other methods from the view points of elapsed time and speed-up on parallel computer with distributed memory.

This paper is organized as follows: In section 2, a short description of BiCGSafe method. In section 3, an explanation of two types of single synchronized BiCGSafe method. In section 4, an explanation of BiCGStar-plus method. In section 5, several results of parallelized iterative methods will be shown, and it will be made clear that the revised single synchronized BiCGSafe methods and BiCGStar-plus method outperform other methods from the view points of elapsed time and speed-up on parallel computer with distributed memory. Finally, in section 6, we have concluding remarks.

Manuscript received April 07, 2015
Seiji Fujino is with Research Institute for Information Technology, Kyushu University, Fukuoka, 812-8581 Japan, e-mail: fujino@cc.kyushu-u.ac.jp
Kosuke Iwasato is a student of Graduate School of Information Science and Electrical Engineering, Kyushu University, Fukuoka, 812-8581 Japan, e-mail: onigili9@gmail.com

II. BiCGSAFE METHOD

The Lanczos polynomial $R_k(\lambda)$ and the auxiliary polynomial $P_k(\lambda)$ satisfy the following two-term recurrence relation as

$$R_0(\lambda) = 1, \quad P_0(\lambda) = 1, \tag{1}$$
$$R_k(\lambda) = R_{k-1}(\lambda) - \alpha_{k-1}\lambda P_{k-1}(\lambda), \tag{2}$$
$$P_k(\lambda) = R_k(\lambda) + \beta_{k-1}P_{k-1}(\lambda), \quad k = 1, 2, \cdots, \tag{3}$$

according to the notation used in ref.[1]. Here, λ means an eigenvalue of a matrix. We introduce the two recurrence parameters ζ_k and η_k. The stabilized polynomial $H_k(\lambda)$ and an auxiliary polynomial $G_k(\lambda)$ satisfy following coupled two-term recurrence as

$$H_0(\lambda) = 1, \; G_0(\lambda) = \zeta_0, \tag{4}$$
$$H_k(\lambda) = H_{k-1}(\lambda) - \lambda G_{k-1}(\lambda), \tag{5}$$
$$G_k(\lambda) = \zeta_k H_k(\lambda) + \eta_k G_{k-1}(\lambda), \quad k = 1, 2, \cdots. \tag{6}$$

We introduce the residual vector $\boldsymbol{r}_k$ as $\boldsymbol{r}_k := H_k(\lambda)R_k(\lambda)\boldsymbol{r}_0$. Here, the vector $\boldsymbol{r}_0$ is the initial residual vector.

The coefficients α_k, β_k can be gained as

$$\alpha_k = \frac{(\boldsymbol{r}_0^*, \; \boldsymbol{r}_k)}{(\boldsymbol{r}_0^*, \; A\boldsymbol{p}_k)}, \tag{7}$$
$$\beta_k = -\frac{(\boldsymbol{r}_0^*, \; A\boldsymbol{t}_k)}{(\boldsymbol{r}_0^*, \; A\boldsymbol{p}_k)} = \frac{\alpha_k(\boldsymbol{r}_0^*, \; \boldsymbol{r}_{k+1})}{\zeta_k(\boldsymbol{r}_0^*, \; \boldsymbol{r}_k)} \tag{8}$$

by the orthogonality conditions $(H_k R_{k+1}\boldsymbol{r}_0, \; \boldsymbol{r}_0^*) = 0$ and $(AH_k P_{k+1}\boldsymbol{r}_0, \; \boldsymbol{r}_0^*) = 0$.

It is known that two parameters ζ_k and η_k are determined by solving the two-dimensional local minimization of the norm of product-type residual $\boldsymbol{r}_{k+1}$ in GPBiCG. However, the residual vector $\boldsymbol{r}_{k+1}$ does not involve both parameters ζ_k, η_k in the update of residual vector. Appearance of another idea needs for overcoming this issue. Therefore, the key idea is to focus on an associate residual vector defined by follow recurence. The associate residual vector $\boldsymbol{a\_r}_k$ can be defined as below.

$$\boldsymbol{a\_r}_k \;\; := \;\; \boldsymbol{r}_k - \zeta_k A\boldsymbol{r}_k - \eta_k \boldsymbol{y}_k. \tag{9}$$

Note that the recurrence (9) is not computed in the iterative loop as it is. We utilize the recurrence (9) only for the recurrence parameters ζ_k and η_k. We call this idea strategy of associate residual.

ISBN: 978-988-19253-4-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

III. Two types of single synchronized BiCGSafe method

A. Single synchronized BiCGSafe version 1

For reduction of synchronization points of BiCGStar method, we improve formulation of parameter β_k as the above mentioned equation (8). For parameter β_k, we can derive alternative expression. $H_k(\lambda)P_{k+1}(\lambda)$ can be written as

$$\begin{aligned} H_k(\lambda)P_{k+1}(\lambda) &= H_k(\lambda)R_{k+1}(\lambda) - \beta_k H_k(\lambda)P_k(\lambda), \\ &= H_k(\lambda)R_k(\lambda) - \alpha_k \lambda H_k(\lambda)P_k(\lambda) \\ &\quad -\beta_k H_k(\lambda)P_k(\lambda). \end{aligned} \tag{10}$$

With the equation (10) and the relation of $(\boldsymbol{x}, A\boldsymbol{y}) = (A^{\mathrm{T}}\boldsymbol{x}, \boldsymbol{y})$, we obtain

$$\begin{aligned} \beta_k &= \frac{(\tilde{\boldsymbol{r}}_0, A(H_k(A)R_k(A)\boldsymbol{r}_0 - \alpha_k A H_k(A)P_k(A)\boldsymbol{r}_0))}{(\tilde{\boldsymbol{r}}_0, AH_k(A)P_k(A)\boldsymbol{r}_0)}, \\ &= \frac{(A^T\tilde{\boldsymbol{r}}_0, H_k(A)R_k(A)\boldsymbol{r}_0) - \alpha_k(A^T\tilde{\boldsymbol{r}}_0, AH_k(A)P_k(A)\boldsymbol{r}_0))}{(\tilde{\boldsymbol{r}}_0, AH_k(A)P_k(A)\boldsymbol{r}_0)}. \end{aligned} \tag{11}$$

Although the equation (11) needs two extra inner products, the coefficients α_k, β_k can be computed at the same place. This means the number of global synchronization points can be reduced. We name this method single synchronized BiCGSafe method version 1(=abbreviated ssBiCGSafe1). We show an algorithm of single synchronized BiCGSafe method as below.

Algorithm 1: ssBiCGSafe1

1. Let $\boldsymbol{x}_0$ be an initial guess, Compute $\boldsymbol{r}_0 = \boldsymbol{b} - A\boldsymbol{x}_0$,
2. Choose $\boldsymbol{r}_0^*$, such that $(\boldsymbol{r}_0^*,\ \boldsymbol{r}_0) \neq 0$,
3. Compute $A^{\mathrm{T}}\boldsymbol{r}_0^*,\ \boldsymbol{y}_0 = \boldsymbol{0},\ \beta_{-1} = 0$,
4. for $k = 0, 1, \dots$ do,
5. Compute $A\boldsymbol{r}_k$,
6. $\boldsymbol{v}_k = \boldsymbol{y}_k + \beta_{k-1}\boldsymbol{u}_{k-1}$,
7. $\boldsymbol{p}_k = \boldsymbol{r}_k + \beta_{k-1}(\boldsymbol{p}_{k-1} - \boldsymbol{u}_{k-1})$,
8. $A\boldsymbol{p}_k = A\boldsymbol{r}_k + \beta_{k-1}(A\boldsymbol{p}_{k-1} - A\boldsymbol{u}_{k-1})$,
9. if $\|\boldsymbol{r}_k\|/\|\boldsymbol{r}_0\| \leq \epsilon$ stop,
10. $\alpha_k = \dfrac{(\boldsymbol{r}_0^*, \boldsymbol{r}_k)}{(\boldsymbol{r}^*{}_0, A\boldsymbol{p}_k)}$,
11. $\beta_k = -\dfrac{(A^{\mathrm{T}}\boldsymbol{r}_0^*, \boldsymbol{r}_k) - \alpha_k(A^{\mathrm{T}}\boldsymbol{r}_0^*, \boldsymbol{p}_k)}{(\boldsymbol{r}_0^*, A\boldsymbol{p}_k)}$,
12. $\zeta_k = \dfrac{(\boldsymbol{y}_k, \boldsymbol{y}_k)(A\boldsymbol{r}_k, \boldsymbol{r}_k) - (A\boldsymbol{r}_k, \boldsymbol{y}_k)(\boldsymbol{y}_k, \boldsymbol{r}_k)}{(A\boldsymbol{r}_k, A\boldsymbol{r}_k)(\boldsymbol{y}_k, \boldsymbol{y}_k) - (A\boldsymbol{r}_k, \boldsymbol{y}_k)(\boldsymbol{y}_k, A\boldsymbol{r}_k)}$,
13. $\eta_k = \dfrac{(A\boldsymbol{r}_k, A\boldsymbol{r}_k)(\boldsymbol{y}_k, \boldsymbol{r}_k) - (A\boldsymbol{r}_k, \boldsymbol{y}_k)(A\boldsymbol{r}_k, \boldsymbol{r}_k)}{(A\boldsymbol{r}_k, A\boldsymbol{r}_k)(\boldsymbol{y}_k, \boldsymbol{y}_k) - (A\boldsymbol{r}_k, \boldsymbol{y}_k)(\boldsymbol{y}_k, A\boldsymbol{r}_k)}$,
    (if $k = 0$ then $\zeta_k = \dfrac{(A\boldsymbol{r}_k, \boldsymbol{r}_k)}{(A\boldsymbol{r}_k, A\boldsymbol{r}_k)},\ \eta_k = 0$)
14. $\boldsymbol{u}_k = \zeta_k A\boldsymbol{p}_k + \eta_k \boldsymbol{v}_k$,
15. Compute $A\boldsymbol{u}_k$,
16. $\boldsymbol{z}_k = \zeta_k \boldsymbol{r}_k + \eta_k \boldsymbol{z}_{k-1} - \alpha_k \boldsymbol{u}_k$,
17. $\boldsymbol{y}_{k+1} = \zeta_k A\boldsymbol{r}_k + \eta_k y_k - \alpha_k A\boldsymbol{u}_k$,
18. $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k + \boldsymbol{z}_k$,
19. $\boldsymbol{r}_{k+1} = \boldsymbol{r}_k - \alpha_k A\boldsymbol{p}_k - \boldsymbol{y}_{k+1}$,
20. end do.

B. Single synchronized BiCGSafe version 2

Single synchronized BiCGSafe method version 1 uses a transposed matrix. In parallel computing, implementation of a transposed matrix vector multiplications is difficult work. Thus, we propose single synchronized BiCGSafe method version 2 without transposed matrix. In our proposed method, the coefficient β_k was computed by the equation (8). We transform the other one coefficient α_k.

In the equation (7), α_k can be written by applying the equation of line 8 in Algorithm 1 as bellow.

$$\alpha_k = \frac{(\boldsymbol{r}_0^*, \boldsymbol{r}_k)}{(\boldsymbol{r}_0^*, A\boldsymbol{p}_k)} = \frac{(\boldsymbol{r}_0^*, \boldsymbol{r}_k)}{(\boldsymbol{r}_0^*, A\boldsymbol{r}_k + \beta_k \boldsymbol{t}_{k-1})}. \tag{12}$$

Although the equation (12) needs two extra inner products, the two coefficients can be computed at the same place as with single synchronized BiCGSafe method version 1. However, our proposed method can compute without a transpose matrix. We show an algorithm of single synchronized BiCGSafe method without a transposed matrix as below. We name this method single synchronized BiCGSafe method version 2(=abbreviated ssBiCGSafe2).

Algorithm 2: ssBiCGSafe2

1. Let $\boldsymbol{x}_0$ be an initial guess,
    Compute $\boldsymbol{r}_0 = \boldsymbol{b} - A\boldsymbol{x}_0$,
2. Choose $\boldsymbol{r}_0^*$ such that $(\boldsymbol{r}_0^*, \boldsymbol{r}_0) \neq 0,\ \beta_{-1} = 0$,
3. **for** $k = 0, 1, \dots$ **do**,
4. Compute $A\boldsymbol{r}_k$,
5. **if** $\|\boldsymbol{r}_k\|/\|\boldsymbol{r}_0\| \leq \epsilon$ **stop**,
6. $\beta_k = \dfrac{\alpha_{k-1}}{\zeta_{k-1}} \dfrac{(\boldsymbol{r}_0^*, \boldsymbol{r}_k)}{(\boldsymbol{r}_0^*, \boldsymbol{r}_{k-1})}$,
7. $\alpha_k = \dfrac{(\boldsymbol{r}_0^*, \boldsymbol{r}_k)}{(\boldsymbol{r}_0^*, A\boldsymbol{r}_k) + \beta_k(\boldsymbol{r}_0^*, \boldsymbol{t}_{k-1})}$,
8. $\zeta_k = \dfrac{(\boldsymbol{y}_k, \boldsymbol{y}_k)(A\boldsymbol{r}_k, \boldsymbol{r}_k) - (A\boldsymbol{r}_k, \boldsymbol{y}_k)(\boldsymbol{y}_k, \boldsymbol{r}_k)}{(A\boldsymbol{r}_k, A\boldsymbol{r}_k)(\boldsymbol{y}_k, \boldsymbol{y}_k) - (A\boldsymbol{r}_k, \boldsymbol{y}_k)(\boldsymbol{y}_k, A\boldsymbol{r}_k)}$,
9. $\eta_k = \dfrac{(A\boldsymbol{r}_k, A\boldsymbol{r}_k)(\boldsymbol{y}_k, \boldsymbol{r}_k) - (A\boldsymbol{r}_k, \boldsymbol{y}_k)(A\boldsymbol{r}_k, \boldsymbol{r}_k)}{(A\boldsymbol{r}_k, A\boldsymbol{r}_k)(\boldsymbol{y}_k, \boldsymbol{y}_k) - (A\boldsymbol{r}_k, \boldsymbol{y}_k)(\boldsymbol{y}_k, A\boldsymbol{r}_k)}$,
    (**if** $k = 0$ **then** $\alpha_k = \dfrac{(\boldsymbol{r}_0^*, \boldsymbol{r}_k)}{(\boldsymbol{r}_0^*, A\boldsymbol{r}_k)},\ \beta_k = 0$,
    $\zeta_k = \dfrac{(A\boldsymbol{r}_k, \boldsymbol{r}_k)}{(A\boldsymbol{r}_k, A\boldsymbol{r}_k)},\ \eta_k = 0$),
10. $\boldsymbol{p}_k = \boldsymbol{r}_k + \beta_k(\boldsymbol{p}_{k-1} - \boldsymbol{u}_{k-1})$,
11. $A\boldsymbol{p}_k = A\boldsymbol{r}_k + \beta_k \boldsymbol{t}_{k-1}$,
12. $\boldsymbol{u}_k = \zeta_k A\boldsymbol{p}_k + \eta_k(\boldsymbol{y}_k + \beta_k \boldsymbol{u}_{k-1})$,
13. Compute $A\boldsymbol{u}_k$,
14. $\boldsymbol{t}_k = A\boldsymbol{p}_k - A\boldsymbol{u}_k$,
15. $\boldsymbol{z}_k = \zeta_k \boldsymbol{r}_k + \eta_k \boldsymbol{z}_{k-1} - \alpha_k \boldsymbol{u}_k$,
16. $\boldsymbol{y}_{k+1} = \zeta_k A\boldsymbol{r}_k + \eta_k \boldsymbol{y}_k - \alpha_k A\boldsymbol{u}_k$,
17. $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k + \boldsymbol{z}_k$,
18. $\boldsymbol{r}_{k+1} = \boldsymbol{r}_k - \alpha_k A\boldsymbol{p}_k - \boldsymbol{y}_{k+1}$,
19. **end do**.

IV. BiCGStar-plus method

BiCGStar-plus method use coupled two-term recurrences of Rutishauser[4] for stabilized polynomial insted of eqations (4)-(6).

$$\begin{cases} \tilde{G}_0(\lambda) = 0, \quad H_0(\lambda) = 1, \\ \tilde{G}_{k+1}(\lambda) = \zeta_k \lambda H_k(\lambda) + \eta_k \tilde{G}_k(\lambda), \\ H_{k+1}(\lambda) = H_k(\lambda) - \tilde{G}_{k+1}(\lambda), \quad k = 0, 1, \dots \end{cases} \tag{13}$$

ISBN: 978-988-19253-4-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

Here, auxiliary polynomial $\tilde{G}_k(\lambda)$ is defined as

$$\tilde{G}_k(\lambda) \quad := \quad H_k(\lambda) - H_{k+1}(\lambda), \quad k = 0, 1, \dots \quad (14)$$

The four coefficients α_k, β_k, ζ_k, η_k can be computed as ssBiCGSafe2 method. Therefore, a synchronizaiton point exists in the algorithm of BiCGStar-plus method. We show an algorithm of BiCGStar-plus method without a transposed matrix as below.

Algorithm 3: BiCGStar-plus

1. Let $\boldsymbol{x}_0$ be an initial guess,
   Compute $\boldsymbol{r}_0 = \boldsymbol{b} - A\boldsymbol{x}_0$,
2. Choose $\boldsymbol{r}_0^*$ such that $(\boldsymbol{r}_0^*, \boldsymbol{r}_0) \neq 0, \ \beta_{-1} = 0$,
3. **for** $k = 0, 1, \dots$ **do**,
4. Compute $A\boldsymbol{r}_k$,
5. **if** $\|\boldsymbol{r}_k\|/\|\boldsymbol{r}_0\| \leq \epsilon$ **stop**,
6. $\beta_k = \dfrac{\alpha_{k-1}}{\zeta_{k-1}} \dfrac{(\boldsymbol{r}_0^*, \boldsymbol{r}_k)}{(\boldsymbol{r}_0^*, \boldsymbol{r}_{k-1})}$,
7. $\alpha_k = \dfrac{(\boldsymbol{r}_0^*, \boldsymbol{r}_k)}{(\boldsymbol{r}_0^*, A\boldsymbol{r}_k) + \beta_k(\boldsymbol{r}_0^*, \boldsymbol{t}_{k-1})}$,
8. $\zeta_k = \dfrac{(\boldsymbol{y}_k, \boldsymbol{y}_k)(A\boldsymbol{r}_k, \boldsymbol{r}_k) - (A\boldsymbol{r}_k, \boldsymbol{y}_k)(\boldsymbol{y}_k, \boldsymbol{r}_k)}{(A\boldsymbol{r}_k, A\boldsymbol{r}_k)(\boldsymbol{y}_k, \boldsymbol{y}_k) - (A\boldsymbol{r}_k, \boldsymbol{y}_k)(\boldsymbol{y}_k, A\boldsymbol{r}_k)}$,
9. $\eta_k = \dfrac{(A\boldsymbol{r}_k, A\boldsymbol{r}_k)(\boldsymbol{y}_k, \boldsymbol{r}_k) - (A\boldsymbol{r}_k, \boldsymbol{y}_k)(A\boldsymbol{r}_k, \boldsymbol{r}_k)}{(A\boldsymbol{r}_k, A\boldsymbol{r}_k)(\boldsymbol{y}_k, \boldsymbol{y}_k) - (A\boldsymbol{r}_k, \boldsymbol{y}_k)(\boldsymbol{y}_k, A\boldsymbol{r}_k)}$,
   (**if** $k = 0$ **then** $\alpha_k = \dfrac{(\boldsymbol{r}_0^*, \boldsymbol{r}_k)}{(\boldsymbol{r}_0^*, A\boldsymbol{r}_k)}, \ \beta_k = 0,$
   $\zeta_k = \dfrac{(A\boldsymbol{r}_k, \boldsymbol{r}_k)}{(A\boldsymbol{r}_k, A\boldsymbol{r}_k)}, \ \eta_k = 0$),
10. $\boldsymbol{s}_k = \boldsymbol{y}_k + \beta_k \boldsymbol{c}_{k-1}$,
11. $\boldsymbol{p}_k = \boldsymbol{r}_k + \beta_k \boldsymbol{w}_{k-1}$,
12. $A\boldsymbol{p}_k = A\boldsymbol{r}_k + \beta_k A\boldsymbol{w}_{k-1}$,
13. $\boldsymbol{v}_k = \zeta_k \boldsymbol{r}_k + \eta_k \boldsymbol{t}_k$,
14. $\boldsymbol{z}_k = \zeta_k A\boldsymbol{r}_k + \eta_k \boldsymbol{y}_k$,
15. $\boldsymbol{c}_k = \zeta_k A\boldsymbol{p}_k + \eta_k \boldsymbol{s}_k$,
16. Compute $A\boldsymbol{c}_k$,
17. $\boldsymbol{w}_k = \boldsymbol{p}_k - \boldsymbol{c}_k$,
18. $A\boldsymbol{w}_k = A\boldsymbol{p}_k - A\boldsymbol{c}_k$,
19. $\boldsymbol{t}_{k+1} = \boldsymbol{v}_k - \alpha_k \boldsymbol{c}_k$,
20. $\boldsymbol{y}_{k+1} = \boldsymbol{z}_k - \alpha_k A\boldsymbol{c}_k$,
21. $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \boldsymbol{w}_k + \boldsymbol{v}_k$,
22. $\boldsymbol{r}_{k+1} = \boldsymbol{r}_k - \alpha_k A\boldsymbol{w}_k - \boldsymbol{z}_{k+1}$,
23. **end do**.

V. Numerical experiments

A. Parallel computational environment and conditions

All computations were done in double precision floating point arithmetic of Fortran90, and performed on Fujitsu PRIMERGY CX400(CPU: Intel Xeon E5-2680, memory: 128Gbytes, OS: Red Hat Linux Enterprise, total nodes: 1476 nodes, cores: 16 cores / 1 node). Fujitsu compiler optimum option "-Kfast" were used. Process parallelization was done by MPI library and OpneMP library. Stopping criterion of iterative methods is less than 10^{-8} of the relative residual 2-norm $||\boldsymbol{r}_{k+1}||_2/||\boldsymbol{b} - A\boldsymbol{x}_0||_2$. In all cases the iteration was started with the initial guess solution $\boldsymbol{x}_0 = (0, 0, \dots, 0)^T$. The initial shadow residual $\boldsymbol{r}_0^*$ is equal to the initial residual $\boldsymbol{r}_0$. Measurement of the elapsed time was done by system function of gettimeofday. All test matrices as shown in Table 1 were normalized with diagonal scaling. Maximum iteration was fixed as 10,000. Number of process varied as 1, 16, 32, 64 and 256. Measurements of the elapsed time per each matrix were five times.

TABLE I
Characteristics of 12 test matrices.

matrix	dimension	nnz	ave. nnz
air-cfl5	1,536,000	19,435,428	12.7
atmosmodd	1,270,432	8,814,880	6.9
poisson3Db	85,623	2,374,949	27.7
raefsky3	21,200	1,488,768	70.2
water_tank	60,740	2,035,281	33.5
circuit5M_dc	3,523,317	19,194,193	5.4
Freescale1	3,428,755	18,920,347	5.5
epb3	84,617	463,625	5.5
sme3Dc	42,930	3,148,656	73.3
thermomech_dK	204,316	2,846,228	13.9
tmt_unsym	917,825	4,584,801	5.0
xenon2	157,464	3,866,688	24.6

In Tables II and III, we demonstrate parallel of Hybrid-version of iterative methods for matrices epb3 and Freescale1, respectively. TRR (True Relative Residual) for the approximated solutions $\boldsymbol{x}_{k+1}$ means $\log_{10}(\|\boldsymbol{b} - A\boldsymbol{x}_{k+1}\|/\|\boldsymbol{b} - A\boldsymbol{x}_0\|)$. Bold figures mean the least elapsed time, and bold speed-ups mean the maximum speed-up.

We can observe the following facts from the results shown in Tables II and III.

1) For matrices epb3 and Freescale1, BiCGStar-plus methods converged fastest as for both the elapsed time and the highest speed-up ratio on 256 processes among the examined iterative methods.
2) For other matrices, the same tendency was gained.

VI. Conclusions

We evaluated Hybrid parallel performance of single synchronized Krylov subspace methods. As a result, we saw that our proposed iterative methods outperformed compared with other methods from the view point of the elapsed time and convergence rate on parallel computer with distributed memory from many numerical examples.

References

[1] Abe, K., Sleijpen, Gerard L.G.: Solving linear equations with a stabilized GPBiCG method, Appl. Numer. Math., Vol.67, pp.4-16, 2013.
[2] Fujino, S., Fujiwara, M., Yoshida, M.: A proposal of preconditioned BiCGSafe method with safe convergence, Proc. of The 17th IMACS World Congress on Scientific Computation, Appl. Math. Simul., CD-ROM, Paris, 2005.
[3] Murakami, K., Fujino, S., A proposal of a product type iterative method using associate residual for parallel computer, Proc. of International workshop on HPC, Krylov Subspace method and its applications, pp.23-26, 2013.
[4] Rutishauser, H.: Theory of gradient method, in: Refined iterative methods for computation of the solution and the eigenvalues of self-adjoint value problems, in: Mitt. Inst. Angew. Math. ETH Zürich, Birkhäuser, pp.24-49, 1959.
[5] University of Florida Sparse Matrix Collection: http://www.cise.ufl.edu/research/sparse/matrices/index.html.
[6] van der Vorst, H.A.: Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems, SIAM J. Sci. Stat. Comput., vol.13, pp.631-644, 1992.
[7] Zhang, S.-L.: GPBi-CG: Generalized product-type methods preconditionings based on Bi-CG for solving nonsymmetric linear systems, SIAM J. Sci. Comput., vol.18, pp.537-551, 1997.

ISBN: 978-988-19253-4-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

TABLE II
PARALLEL PERFORMANCE OF HYBRID-VERSION OF ITERATIVE METHODS FOR MATRIX EPB3.

method	np	Mv	tot.time [sec.]	ratio	ave.time [msec.]	speed-up	TRR
GPBiCG	1	3,852	7.083	1.00	1.839	1.00	-8.0
	16	4,056	0.509	1.00	0.125	14.65	-8.0
	32	3,798	0.289	1.00	0.076	24.17	-8.0
	64	4,178	0.211	1.00	0.051	36.41	-8.0
	128	3,998	0.162	1.00	0.041	45.38	-8.0
	256	4,114	0.174	1.00	0.042	43.48	-8.0
GPBiCG_v4	1	3,816	7.618	1.08	1.996	1.00	-8.0
	16	3,630	0.428	0.84	0.118	16.93	-8.0
	32	3,854	0.275	0.95	0.071	27.98	-8.0
	64	3,838	0.183	0.87	0.048	41.87	-8.0
	128	3,594	0.145	0.90	0.040	49.48	-8.0
	256	3,998	0.144	0.83	0.036	55.43	-8.0
BiCGSafe	1	3,764	6.172	0.87	1.640	1.00	-8.0
	16	3,770	0.415	0.82	0.110	14.90	-8.0
	32	3,464	0.233	0.81	0.067	24.38	-8.0
	64	3,956	0.179	0.85	0.045	36.24	-8.0
	128	3,822	0.134	0.83	0.035	46.77	-8.0
	256	3,684	0.130	0.75	0.035	46.47	-8.0
ssBiCGSafe1	1	3,910	6.632	0.94	1.696	1.00	-8.0
	16	3,870	0.427	0.84	0.110	15.37	-8.0
	32	3,746	0.259	0.90	0.069	24.53	-8.0
	64	3,858	0.163	0.77	0.042	40.15	-8.0
	128	3,616	0.112	0.69	0.031	54.76	-8.0
	256	3,844	0.106	0.61	0.028	61.51	-8.0
ssBiCGSafe2	1	3,716	6.320	0.89	1.701	1.00	-8.0
	16	3,708	0.410	0.81	0.111	15.38	-8.0
	32	3,706	0.241	0.83	0.065	26.15	-8.0
	64	3,518	0.143	0.68	0.041	41.84	-8.0
	128	3,950	0.118	0.73	0.030	56.93	-8.0
	256	3,722	**0.101**	0.58	0.027	62.68	-8.0
BiCGStar-plus	1	3,548	6.706	0.95	1.890	1.00	-8.0
	16	4,132	0.453	0.89	0.110	17.24	-8.0
	32	3,872	0.257	0.89	0.066	28.48	-8.0
	64	3,836	0.164	0.78	0.043	44.21	-8.0
	128	3,726	0.116	0.72	0.031	60.71	-8.1
	256	3,754	**0.101**	0.58	0.027	**70.25**	-8.0

TABLE III
PARALLEL PERFORMANCE OF HYBRID-VERSION OF ITERATIVE METHODS FOR MATRIX FREESCALE1.

method	np	Mv	tot.time [sec.]	ratio	ave.time [msec.]	speed up	TRR
GPBiCG	1	9,534	914.699	1.00	95.941	1.00	-8.0
	16	9,328	265.123	1.00	28.422	3.38	-8.0
	32	9,370	168.007	1.00	17.930	5.35	-8.0
	64	9,604	97.763	1.00	10.179	9.42	-8.0
	128	9,536	54.012	1.00	5.664	16.94	-8.0
	256	9,120	27.708	1.00	3.038	31.58	-8.0
GPBiCG_v4	1	9,682	1005.108	1.10	103.812	1.00	-8.0
	16	9,772	274.881	1.04	28.129	3.69	-8.0
	32	9,474	161.168	0.96	17.012	6.10	-8.0
	64	9,754	93.241	0.95	9.559	10.86	-8.0
	128	9,468	52.103	0.96	5.503	18.86	-8.0
	256	9,042	26.029	0.94	2.879	36.06	-8.0
BiCGSafe	1	9,558	862.245	0.94	90.212	1.00	-8.0
	16	9,634	247.121	0.93	25.651	3.52	-8.0
	32	10,342	168.452	1.00	16.288	5.54	-8.0
	64	9,152	85.350	0.87	9.326	9.67	-8.0
	128	9,364	50.858	0.94	5.431	16.61	-8.0
	256	9,198	26.757	0.97	2.909	31.01	-8.0
ssBiCGSafe1	1	9,380	854.881	0.93	91.139	1.00	-8.0
	16	9,386	243.198	0.92	25.911	3.52	-8.0
	32	9,360	148.453	0.88	15.860	5.75	-8.0
	64	8,926	82.751	0.85	9.271	9.83	-8.0
	128	8,962	45.679	0.85	5.097	17.88	-8.0
	256	9,202	25.594	0.92	2.781	32.77	-8.0
ssBiCGSafe2	1	9,618	846.817	0.93	88.045	1.00	-8.0
	16	9,070	233.475	0.88	25.741	3.42	-8.0
	32	10,662	168.890	1.01	15.840	5.56	-8.0
	64	9,986	91.949	0.94	9.208	9.56	-8.0
	128	10,058	50.498	0.93	5.021	17.54	-8.0
	256	9,562	25.959	0.94	2.715	32.43	-8.0
BiCGStar-plus	1	9,066	925.135	1.01	102.044	1.00	-8.0
	16	9,488	248.236	0.94	26.163	3.90	-8.0
	32	9,088	144.791	0.86	15.932	6.40	-8.0
	64	9,548	88.443	0.90	9.263	11.02	-8.0
	128	10,156	51.498	0.95	5.071	20.12	-8.0
	256	9,264	**25.295**	0.91	2.730	**37.37**	-8.0

ISBN: 978-988-19253-4-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)