Communication-Hiding Pipelined BiCGStar-Plus Method and Its Application to GPU-based Numerical Simulation of Blood Flow

Viet Huynh Quang Huy
Hiroshi Suito
hqhviet@tohoku.ac.jp
hiroshi.suito@tohoku.ac.jp
Advanced Institute for Materials Research, Tohoku University
Sendai, Miyagi, Japan

Algorithm 1 Pipelined BiCGStar-plus

Let $\boldsymbol{x}_0$ is an initial guess,
Compute $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$,
Choose $\boldsymbol{r}_0^*$ such that $(\boldsymbol{r}_0^*, \boldsymbol{r}_0) \neq 0$, e.g., $\boldsymbol{r}_0^* = \boldsymbol{r}_0$,
Initialize $\boldsymbol{e}_0 = \boldsymbol{A}\boldsymbol{r}_0^*$,
Initialize $\boldsymbol{c}_0 = \boldsymbol{d}_0 = \boldsymbol{f}_0 = \boldsymbol{g}_0 = \boldsymbol{h}_0 = \boldsymbol{l}_0 = \boldsymbol{o}_0 = \boldsymbol{p}_0 = \boldsymbol{0}$,
Initialize $\boldsymbol{q}_0 = \boldsymbol{s}_0 = \boldsymbol{t}_0 = \boldsymbol{u}_0 = \boldsymbol{v}_0 = \boldsymbol{w}_0 = \boldsymbol{y}_0 = \boldsymbol{z}_0 = \boldsymbol{0}$,
for $i = 0, 1, \dots$ **do**
  if $||\boldsymbol{r}_i||/||\boldsymbol{r}_0|| \leq \epsilon$ **stop**,
  Define $\rho_i := (\boldsymbol{r}_0^*, \boldsymbol{r}_i)$, $\sigma_i := (\boldsymbol{r}_0^*, \boldsymbol{e}_i)$, $\tau_i := (\boldsymbol{r}_0^*, \boldsymbol{u}_i)$,
  Define $\mu_i := (\boldsymbol{e}_i, \boldsymbol{e}_i)$, $\nu_i := (\boldsymbol{e}_i, \boldsymbol{r}_i)$, $\lambda_i := (\boldsymbol{y}_i, \boldsymbol{y}_i)$,
  Define $\omega_i := (\boldsymbol{e}_i, \boldsymbol{y}_i)$, $\xi_i := (\boldsymbol{y}_i, \boldsymbol{r}_i)$,
  if $i = 0$ **then**
    $\beta_i = 0, \alpha_i = \rho_i/\sigma_i$,
    $\zeta_i = \nu_i/\mu_i, \eta_i = 0$,
  else
    $\beta_i = (\alpha_{i-1}/\zeta_{i-1})(\rho_i/\rho_{i-1})$,
    $\alpha_i = \rho_i/(\sigma_i + \beta_i\tau_i)$,
    $\zeta_i = (\lambda_i\nu_i - \omega_i\xi_i)/(\mu_i\lambda_i - \omega_i^2)$,
    $\eta_i = (\mu_i\xi_i - \omega_i\nu_i)/(\mu_i\lambda_i - \omega_i^2)$,
  end if
  Compute $\boldsymbol{A}\boldsymbol{e}_i$,
  $\boldsymbol{s}_i = \boldsymbol{y}_i + \beta_i\boldsymbol{c}_{i-1}$,
  $\boldsymbol{f}_i = \boldsymbol{g}_i + \beta_i\boldsymbol{d}_{i-1}$,
  $\boldsymbol{p}_i = \boldsymbol{r}_i + \beta_i\boldsymbol{w}_{i-1}$,
  $\boldsymbol{q}_i = \boldsymbol{e}_i + \beta_i\boldsymbol{u}_{i-1}$,
  $\boldsymbol{o}_i = \boldsymbol{A}\boldsymbol{e}_i + \beta_i\boldsymbol{l}_{i-1}$,
  $\boldsymbol{v}_i = \zeta_i\boldsymbol{r}_i + \eta_i\boldsymbol{t}_i$,
  $\boldsymbol{z}_i = \zeta_i\boldsymbol{e}_i + \eta_i\boldsymbol{y}_i$,
  $\boldsymbol{h}_i = \zeta_i\boldsymbol{A}\boldsymbol{e}_i + \eta_i\boldsymbol{g}_i$,
  $\boldsymbol{c}_i = \zeta_i\boldsymbol{q}_i + \eta_i\boldsymbol{s}_i$,
  $\boldsymbol{d}_i = \zeta_i\boldsymbol{o}_i + \eta_i\boldsymbol{f}_i$,
  Compute $\boldsymbol{A}\boldsymbol{d}_i$,
  $\boldsymbol{w}_i = \boldsymbol{p}_i - \boldsymbol{c}_i$,
  $\boldsymbol{u}_i = \boldsymbol{q}_i - \boldsymbol{d}_i$,
  $\boldsymbol{l}_i = \boldsymbol{o}_i - \boldsymbol{A}\boldsymbol{d}_i$,
  $\boldsymbol{t}_{i+1} = \boldsymbol{v}_i - \alpha_i\boldsymbol{c}_i$,
  $\boldsymbol{y}_{i+1} = \boldsymbol{z}_i - \alpha_i\boldsymbol{d}_i$,
  $\boldsymbol{g}_{i+1} = \boldsymbol{h}_i - \alpha_i\boldsymbol{A}\boldsymbol{d}_i$,
  $\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \boldsymbol{v}_i + \alpha_i\boldsymbol{w}_i$,
  $\boldsymbol{r}_{i+1} = \boldsymbol{r}_i - \boldsymbol{z}_i - \alpha_i\boldsymbol{u}_i$,
  $\boldsymbol{e}_{i+1} = \boldsymbol{e}_i - \boldsymbol{h}_i - \alpha_i\boldsymbol{l}_i$,
end for

ABSTRACT

One of the widely used methods for solving linear equation systems is the BiCGStab [4] iterative algorithm, which uses an initial solution and creates a sequence of improved approximate solutions. The BiCGStab algorithm is built from three basic operations: inner-product, linear combination (the addition of a scalar multiple of one vector to another vector), and matrix-vector product. In the parallel implementation of the BiCGStab algorithm, one main problem that causes delays to the whole process is the inner product operation, which requires a global synchronization phrase for one global communication operation to collect the scalar partial sums in each processor to one processor, and one global communication operation for distributing the result to all processors. Time for inner product computation will dominate the time of the whole algorithm as the number of processors increases. Recently, the new variants of the BiCGStab algorithm with hiding communication latency of computing inner products by overlapping inner-product computations with a matrix-vector computation have been proposed by Cools and Vanroose [1]. On parallel computers, this method can gain higher scalability property than the standard BiCGStab method. Among generalized algorithms of the BiCGStab method such as GPBiCG, BiCGStar-plus, BiCGStar-plus has good convergence behavior. In this poster, similar to the work of Cools and Vanroose, we propose a variant of BiCGStar-plus named Pipelined BiCGStar-plus that hides communication latency. To verify the effectiveness of the proposed algorithm for real problems, we apply it to blood flow simulation.

KEYWORDS

Parallelization, Latency hiding, Krylov subspace methods, Generalized product-type Bi-CG method, BiCGStar-plus method.

REFERENCES

[1] S. Cools and W. Vanroose. 2016. The communication-hiding pipelined BiCGStab method for the efficient parallel solution of large unsymmetric linear systems. *Parallel Comput.* 65 (12 2016). https://doi.org/10.1016/j.parco.2017.04.005

[2] S. Fujino and K. Murakami. 2013. A Parallel Variant of BiCGStar-Plus Method Reduced to Single Global Synchronization. In *AsiaSim 2013*. Springer, Berlin, Heidelberg, 325–332.

[3] H. Suito V. Q. H. Huynh. 2016. Multi-GPU Implementation of a Parallel Solver for In-compressible Navier-Stokes Equations Discretized by Stabilized Finite Element Formulations. *Kokyuroku* 2037 (2016), 149–152.

[4] H. A. van der Vorst. 1992. Bi-CGSTAB: A Fast and Smoothly Converging Variant of Bi-CG for the Solution of Nonsymmetric Linear Systems. *SIAM J. Scientific Computing* 13 (1992), 631–644.

[5] S. L. Zhang. 1997. GPBi-CG: Generalized Product-type Methods Based on Bi-CG for Solving Nonsymmetric Linear Systems. *SIAM J. Scientific Computing* 18 (1997), 537–551.