GPU Implementation of Splitting-Up Conjugate Gradient Method

Akiyoshi Wakatani (Konan University, Kobe, JAPAN)

Summary: This paper describes the implementation of a preconditioned CG (Conjugate Gradient) method on GPUs and evaluates the performance compared with CPUs. Our CG method utilizes SP (Splitting-Up) preconditioner [1], which is suitable for parallel processing because other dimensions except for one dimension are independent. In order to enhance the memory bandwidth to the global memory of the GPU, our implementation utilizes a pseudo matrix transposition before and after a tridiagonal matrix solver, which results in coalesced memory accesses. By this policy, the speedups of our approach can be enhanced by up to 91%.

What is Splitting-Up Conjugate Gradient method?

SPCG (Splitting-Up Conjugate Gradient) method is one of preconditioned CG methods and it can be easily parallelized. According to the following paper, the speed of convergence of SPCG is comparable to that of ICCG. They compare SPCG (SPBiCG) with ICCG (ILUBiCG) and show that the SP method exhibits the small increase of 20-30% in the number of iterations.

[1] S. Odanaka, T. Nogi, "Massively parallel computation using a splitting-up operator method for three-dimensional device simulation," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems. vol. 14, issue 7, pp. 824-832, 1995.

Background and proposed method

The Main loop of SPCG (Splitting-Up Conjugate Gradient) method for $\mathbf{Au} = \mathbf{f}$.

$$\mathbf{h}^n = \mathbf{C}^{-1}(\mathbf{Au}^n - \mathbf{f})$$

$$\tau = \frac{(\mathbf{h}^n, \mathbf{Ch}^n)}{(\mathbf{d}^n, \mathbf{Ad}^n)}$$

$$\mathbf{u}^{n+1} = \mathbf{u}^{n+1} + \tau \mathbf{d}^n$$

$$\mathbf{h}^{n+1} = \mathbf{h}^{n+1} + \tau \cdot \boxed{\mathbf{C}^{-1}} \mathbf{Ad}^n$$

$$\beta = \frac{(\mathbf{h}^{n+1}, \mathbf{Ch}^{n+1})}{(\mathbf{h}^n, \mathbf{Ch}^n)}$$

$$\mathbf{d}^{n+1} = -\mathbf{h}^{n+1} + \beta \mathbf{d}^n$$

$\mathbf{h}^n$: residual vector $\mathbf{u}^n$: numerical solution
$\mathbf{C}^{-1}$: predonditioner $\mathbf{d}^n$: direction vector

SP preconditioner for 3D Laplace equation

$$C = (D+X)D^{-1}(D+Y)D^{-1}(D+Z)$$

$$D^{-1} = [(0,0,0)(0,(0,1/6,0),0)(0,0,0)]$$

$$(D+X) = [(0,0,0)(0,(-1,6,-1),0)(0,0,0)]$$

$$(D+Y) = [(0,0,0)((0,-1,0),(0,6,0),(0,-1,0))(0,0,0)]$$

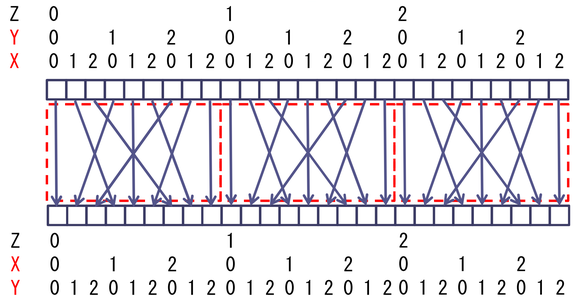
$$(D+Z) = [(0,(0,-1,0),0)(0,(0,6,0),0)(0,(0,-1,0),0)]$$

SP preconditioner for 3D Laplace equation consists of three tridiagonal matrix solvers, which are solved by Thomas method that is composed of a forward substitution and a backward substitution.

However, the solver in x-direction is done with non-coalesced memory accesses, which leads to a severe performance degradation.

Pseudo matrix transposition

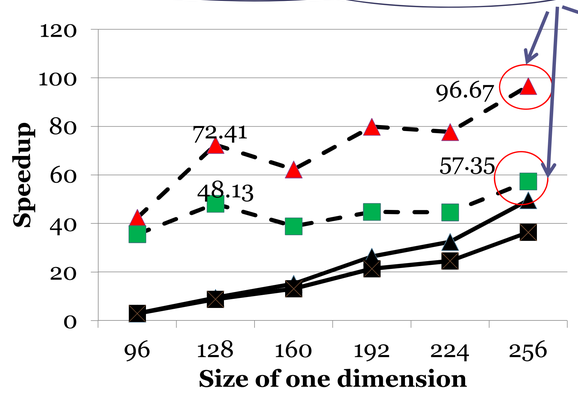
In order to avoid non-coalesced memory accesses, a pseudo matrix transposition should be carried out before and after the x-direction solver.



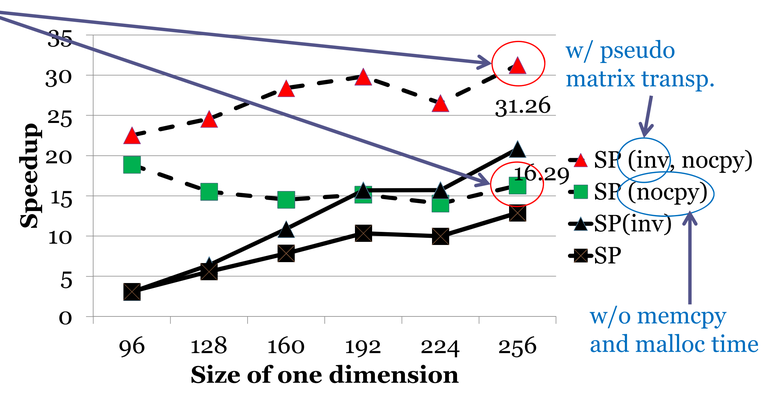
On x-direction solver, adjacent threads in a thread block are assigned to adjacent elements in y-direction. So, memory accesses in the forward substitution and the backward substitutions can be coalesced.

Results and discussion

By using pseudo matrix transpositions, the performance can be enhanced by up to 91%.

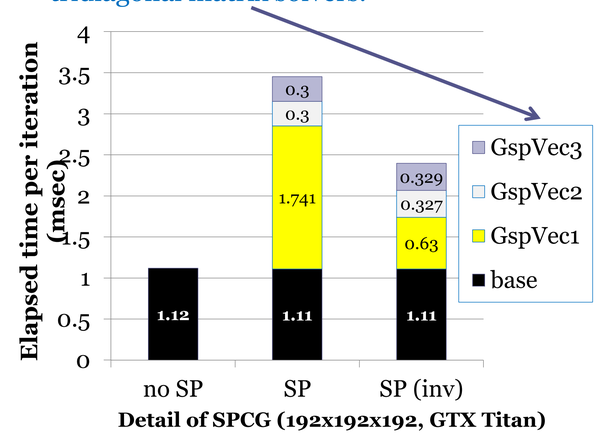


PC1:
GTX Titan+ Intel Corei5-3470 CPU (3.20GHz)
with OpenMP multithreading (4 threads)



PC2:
Tesla K80 + XeonCPU E5-2650 v3 (2.30GHz)
with OpenMP multithreading (10 threads)

"base" is the time of the CG method, and "GspVecX" is the time of one of three tridiagonal matrix solvers.



By applying the pseudo matrix transposition to GspVec1 routine that originally includes non-coalesced memory accesses, the time of GspVec1 can be dramatically reduced, even though the time of the transposition is added.

In the left figure, two pseudo matrix transpositions are carried out before and after GspVec1. However, the number of the transpositions can be reduced by a rotation configuration. We will evaluate the effectiveness in the future.

In the future, we will compare the performance of the SPCG method with that of a parallelized ICCG method on GPUs and we will also evaluate the speed of convergence of both methods. In addition, the number of pseudo matrix transpositions can be reduced to one by using a rotation configuration technique.

甲南大学 KONAN UNIVERSITY
Contact:
Akiyoshi Wakatani (Konan University, Faculty of Intelligence and Informatics) Email: wakatani @konan-u.ac.jp FAX +81-78-435-2540
KONAN UNIVERSITY