Acceleration of Numerical Turbine using the Red-Black Method

Yuta Hougi, Kazuhiko Komatsu, Osamu Watanabe, Masayuki Sato, Hiroaki Kobayashi
Tohoku University, Sendai, Miyagi, Japan
{yuta.hougi.t8@dc.,komatsu@,masa@,koba@}tohoku.ac.jp

1 INTRODUCTION

Numerical Turbine is a simulation code to analyze the flows in steam turbines for the design of highly-efficient turbines and to analyze mechanisms of blades deterioration [1]. Since the scales of simulations have been expanded in recent years in order to simulate the flows more accurately, it is necessary to shorten the execution time. One of the most dominant routines, called *implicit*, occupies 35% of the total execution time. Thus, it is important to reduce this routine time. The implicit routine is optimized by the hyperplane method [2] to exploit vector computing units in most of recent processors. However, the hyperplane method requires memory accesses with large strides, and their long memory access time is an obstacle to further acceleration. In this poster, we implement the Red-Black method, and discuss its effectiveness through experimental results quantitatively.

2 CALCULATION IN IMPLICIT ROUTINE

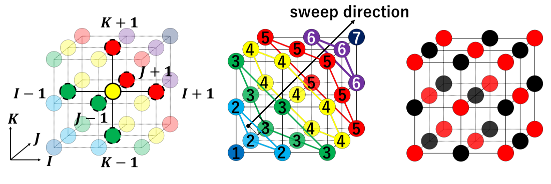
The implicit routine calculates physical fields ΔQ such as momentum and internal energy of mass unit derived from the vector of flux, the viscous term, and the source term of the Navier-Stokes equations. Figure 1(a) shows that the adjacent points are referenced when calculating ΔQ of the center grid point. The center point and its neighbors are not calculated at the same time to avoid updating ΔQ in the irregular order. Therefore, the calculations of the grid points cannot be easily vectorized.

Figure 1(b) shows how the implicit routine is vectorized by using the hyperplane method. The numbers in the grid points indicate the order of a sweep. A group of grid points with the same number is called a *hyperplane*. In the calculations of the grid points in the m-th hyperplane, their adjacent points exist in the $(m-1)$ and $(m+1)$-th hyperplanes. This allows vector calculation of grid points in the m-th hyperplane by avoiding dependency among the grid points and their neighbors.

However, the vector calculation by the hyperplane method requires large stride memory accesses. From the view point of the data layout in the memory, the grid points on the same hyperplane are arranged with strides of $I_{size}-2$, and they increase as the problem size becomes large. Since a processor accesses data at block granularity, large stride memory accesses increase the number of loaded blocks. As the result, the memory access time becomes a bottleneck.

3 RED-BLACK METHOD

Figure 1(c) shows the Red-Black method. All the grid points are color-coded in the order of red and black. At first, all the red points are calculated, and then all the black points are calculated. In the calculation at the n-th step, ΔQ^n of the red points are calculated using ΔQ^{n-1} of the adjacent black points, and then ΔQ^n of the black points are updated by using ΔQ^n of the adjacent red points. Since the Red-Black method avoids data dependency among the adjacent points, the calculations of the points in a group of the same color can be vectorized.



(a) Referenced neighbors (b) Hyperplane (c) Red-Black

Figure 1: Vectorization methods.

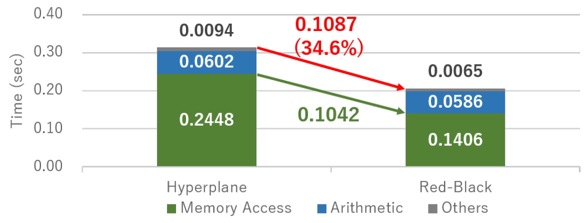


Figure 2: Execution time using two methods.

Since the red and black points are arranged alternately in the memory, strides for each color are fixed to 1 regardless of the problem scale, which are much shorter than those of the hyperplane method. As a result, the number of memory accesses can be reduced compared to the hyperplane method.

4 EVALUATION

In this evaluation, we use NEC SX-Aurora TSUBASA Vector Engine Type 10B. Figure 2 shows the execution time of the implicit kernel vectorized by using the hyperplane method and the Red-Black method. The results show that the memory access time is reduced by 42.6%. As a result, the Red-Black method can reduce the execution time by 34.6% compared to the hyperplane method. While the number of memory accesses in the hyperplane method is 3.43×10^8, that in the Red-Black method is 1.97×10^8. Therefore, the Red-Black method can accelerate the implicit routine by reducing the number of memory accesses in addition to shortening memory access strides.

5 CONCLUSION

In this poster, we discuss the Red-Black method for the implicit routine to reduce the number of memory accesses. The evaluation results show that the Red-Black method can reduce the execution time of the routine by 34.6%. As future work, we plan to investigate the effect of acceleration by improving the cache hit rate by cache blocking for the implicit routine with the Red-Black method.

REFERENCES

[1] S. Yamamoto et al., "Parallel computation of condensate flows through 2-d and 3-d multistage turbine cascades." In *Proc. Intl. Gas Turbine Congress*, 2007.
[2] H. Matsuoka et al., "Program optimization of Numerical Turbine for vector supercomputer SX-ACE." In *Proc. Parallel CFD2016*, p. 8, 2017.