Performance evaluation of multilevel Parareal method

Koki Hiromi, Akihiro Fujii, Teruo Tanaka
Kogakuin University

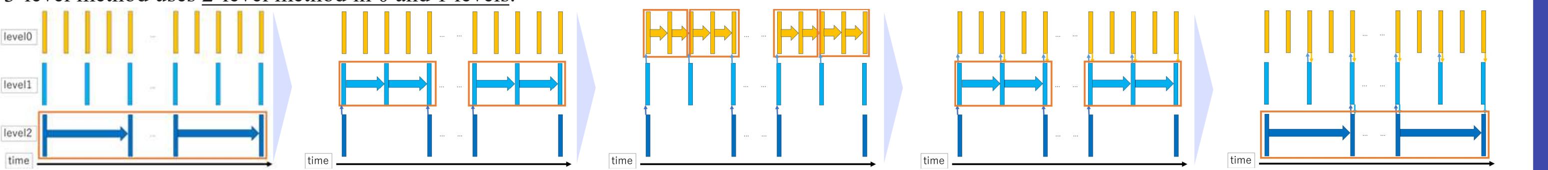
KOGAKUIN UNIVERSITY | KUTE-TOKYO Kogakuin University of Technology & Engineering

Introduction

- Parallel in Time is gaining attention because it brings out the computer performance by introducing the new dimension in parallelization.
  Parareal method is one of the popular parallel in time algorithms.
  - Iterative method consisting of sequential coarse and parallel fine solutions.
  - For large-scale problems, the time for the sequential coarse solution tend to occupy the most part of the execution time.
    This problem can be solved by increasing the number of levels.
    Several methods are known[1][2] to implement multilevel Parareal methods, but their effectiveness has not been shown as far as the authors know.
- In this study, we prepare the 3-level Parareal method with OpenMP nested and evaluate the 3-level Parareal method[1].

Parareal method

Repeat the following flow.
3-level method uses 2-level method in 0 and 1 levels.

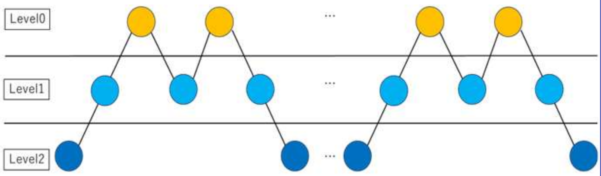


Sequential Coarse solver (level-2) Parallel Coarse solver (level-1) Parallel Fine solver (level-0) Correction Coarse solution (level-1) Correction Coarse solution (level-2)

Convergence condition

- Parareal method corrects and reduces errors between coarse and fine solutions through iteration.
- Convergence threshold was set by a relative error with respect to the last time step solution solved by a explicit method
  (x_c: fine solution at last timestep, x: Parareal method solution at last timestep).

$$r = \frac{||x_c - x||^2}{|x_c|^2}$$



Operation transition between levels in 3-level Parareal method

Numerical experiment

Target problem : one-dimension thermal diffusion problem based on an explicit method. Wisteria-O at the univ. of Tokyo was used.

Timesteps per level(small-size)

levels	2level method	3level method
0	1000	1000
1	100	100
2	-	10

Threads per level(small-size)

levels	2level method	3level method
0	1-50	1-50
1	1	1-5
2	-	1

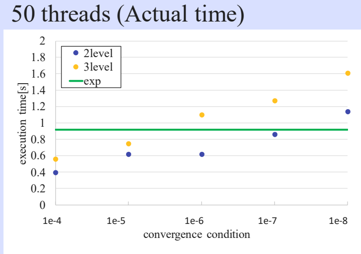
Timesteps per level(large-size)

level	2level method	3level method
0	100000	100000
1	10000	10000
2	-	100

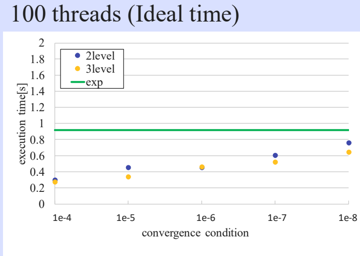
Threads per level(large-size)

levels	2level method	3level method
0	1-50	1-50
1	1	1-50
2	-	1

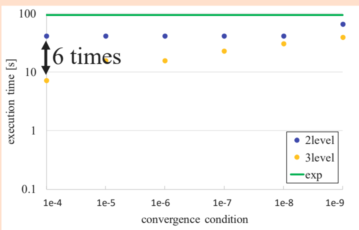
50 threads (Actual time)



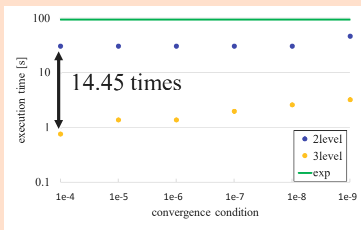
100 threads (Ideal time)



50 threads (Actual time)



10000 threads (Ideal time)



- **Ideal time**:
  It is estimated with the calculation cost in one time step at level-0 and the degree of parallelism for each level.
- The 3-level method was faster than the 2-level method in the large-sized problem.
  - For ideal time, the 3-level method is faster than the 2-level method for small problems.
  - With complete parallelism, the 3-level method was faster than the explicit method regardless of the convergence condition.
- The 3-level method was up to about 6 times faster than the 2-level method in large size problem with actual time.

Conclusion

- Even if the convergence conditions were raised, the three-level method with OpenMP was effective for large size problems.
- Ideal time comparison reveal the usefulness of the three-level method.
  - For both problems, if the limit on parallelism in computing environments was removed in the ideal time, the 3-level method was able to reduce the execution time.
  - The 3-level method is expected to be 1.18 times faster on small problems and 14.45 times faster on large problems than the 2-level method in ideal time with complete parallelism.

Acknowledgement

This work is supported by JSPS KAKENHI Grant Number JP22K12063, “Joint Usage/Research Center for Interdisciplinary Large-scale Information Infrastructures” and “High Performance Computing Infrastructure” in Japan (Project ID: jh230060-NAH).

Reference

[1] Rosemeier, Juliane, Terry Haut, and Beth Wingate. "Multi level Parareal algorithm with Averaging." arXiv preprint arXiv:2211.17239 (2022).
[2] Y. -C. Chen and K. Nakajima, “Optimized Cascadic Multigrid Parareal Method for Explicit Time-Marching Schemes,” 2021 12th Workshop on Latest Advances in Scalable Algorithms for Large-Scale Systems (ScalA), St. Louis, MN, USA, 2021, pp. 9-18, doi: 10.1109/ScalA54577.2021.00007.