2 January 2025

Review of "asQ: parallel-in-time finite element simulations using ParaDiag for geoscientific models and beyond".

This paper is a well-written overview of the ParaDiag-II and related parallel-in-time methods. Its primary focus is the implementation of ParaDiag-II in scalable Python and PETSc frameworks and the validation of this implementation using four example problems. I recommend the article be accepted with minor revisions. Following are some points to consider in revisions as well as some typos to fix.

1. "with a survey [of] previous research"
2. "in Schreiber et al. (2018); Schreiber and Loft (2019); Schreiber et al. (2019) used a related approach with coefficients": missing punctuation
3. Missing period after eq. 25.
4. Comma rather than period after eq. 26.
5. It's a matter of taste, but one could imagine making T_c and T_b independent of k_p, thus making T_c a measure of the communication cost per application of P and similarly for T_b. This construction seems to align better with the other cost measures. The final line in equation 30 would still come out the same since k_p would cancel. Feel free to reject this suggestion; I make it in case it had not occurred to you.
6. "$T_c << T_b$": use `\ll` ($\ll$) rather than `<<`.
7. "the convergence criteria is": "criterion", singular
8. "ARCHER2 consists of 5860 nodes, each with 2 AMD EPYC 7742 CPUs": I suggest some terminology be clarified. I conclude there are $2 \cdot 5860 = 11720$ CPUs total on the machine. My understanding of usage is that "CPU" = "processor". But Fig. 3 shows number of "processors" out to $16384 > 11720$, and Fig. 6 shows 32768. Now I'm wondering if the horizontal axis in Fig. 3 is actually number of cores, not number of CPUs. If so, is my usage incorrect? By "processor" do most people mean "core" and not "CPU"?
9. "A quadrilateral mesh with 128^2 elements is used resulting in ∼65kDoFs, which is small enough to fit on a single core for the serial-in-time method." I'm not understanding. Is only one core used for the serial-in-time method?
10. These seem in conflict (64 vs 32 cores/node):
    - "the best performance was obtained by underfilling each node by allocating only two cores per L3 cache. This strategy is used in all examples unless otherwise stated, giving a maximum of 64 cores per node."
    - "It was found that the best results were obtained by allocating only a single core per L3 cache up to a maximum of 32 cores per node."

Or does this apparent discrepancy have something to do with the complex-valued blocks?

11. Fig. 9: In the legend, I think "$N\Delta t$" should be "$N_t\Delta t$".

12. Fig. 13: Minor grid lines are on, unlike in the other figures. I like minor grid lines, but only if they are a much lighter shade; these are a bit obtrusive. I suggest either adding minor grid lines to all the figures or removing them from this one. If the former, consider using `pyplot.grid` options to make thin, light-gray lines that are very faint compared with the rest of the plot entities.

13. Fig. 13: Consider reducing the y limits, especially at the top of the plot.

14. Fig. 14: The "1000" in the top-left corner seems to have a rendering problem.

15. "methods have been recently been implemented": Fix typo.

16. The numerical experiments are well presented and very interesting. I have one request. It is unclear to me that the block equations are being solved at the practical strong-scaling limit, i.e., space parallelism has been saturated, following the terminology in the first sentence of Sect. 3.3. I believe nonspecialist readers, an I am one, would benefit from one additional plot, probably for the nonlinear shallow water case, that shows the serial-in-time simulation at each resolution strong-scaled out in number of processors until speedup is lost. That is, take mesh 7, for example, and run the serial-in-time simulation on a range of processor counts, possibly up through all available processors. That way, the reader can gain some intuition about the following question: If I have N processors available to me, how should I configure my simulation in terms of N_t vs parallelism in space? An alternative is to clarify for the reader that one of two situations holds in the experiments: either (1) space parallelism is indeed saturated in the serial-in-time runs or (2) for the sake of tractability of experiments, you're pretending it is.

17. There is a fair bit of python code in this manuscript. If GMD permits it, it would be nice to use the Latex listings package or an alternative rather than verbatim.