The 8th European Congress on Computational Methods in Applied Sciences and Engineering
ECCOMAS Congress 2022
5-9 June 2022, Oslo, Norway

Cleaning up distributed objects in managed languages and applications in extremely large scale simulations

Jack Betteridge*[1], Patrick Farrell[2] and David Ham[3]

[1] Imperial College London, Huxley Building, South Kensington Campus, London, SW7 2AZ j.betteridge@imperial.ac.uk, www.imperial.ac.uk/people/j.betteridge
[2] University of Oxford, Andrew Wiles Building, Radcliffe, Observatory Quarter, Woodstock Road, Oxford, OX2 6GG, patrick.farrell@maths.ox.ac.uk, www.maths.ox.ac.uk/people/patrick.farrell
[3] Imperial College London, Huxley Building, South Kensington Campus, London, SW7 2AZ, david.ham@imperial.ac.uk, www.imperial.ac.uk/people/david.ham

Keywords: *Partial Differential Equations, Finite Element Method, Code Generation, Exascale*

Memory managed languages such as Python and Julia are being used increasingly often on high performance computers (HPC) to drive extremely large, pre-exascale simulations.

Using Firedrake, a code generation framework for solving partial differential equations (PDEs), we can demonstrate that generating simulation code from a high-level Python interface provides an effective mechanism for creating high performance simulations from very few lines of user code [1]. Advantages of working in a managed language include the flexibility to change discretisations and solvers for better hardware utilisation, as well as being more productive for scientists and engineers. However, one drawback is that the memory management of such a language can create havoc when attempting to clean up distributed objects.

When running Firedrake in parallel, it is possible for Python's garbage collector to cause a deadlock when attempting to clean up distributed PETSc data structures that require collective destruction. Turning off Python's garbage collection is a poor workaround at best and a catastrophic memory leak at worst.

We outline an algorithm for the safe parallel destruction of distributed objects that can be used in any managed language and assess its impact on performance using different HPC facilities using a reference implementation in PETSc/petsc4py. Furthermore, we demonstrate some of the new very large scale simulations written using Firedrake that these improvements allow us to run.

REFERENCES

[1] J. D. Betteridge, P. E. Farrell and D. A. Ham, *Code Generation for Productive, Portable, and Scalable Finite Element Simulation in Firedrake* Computing in Science & Engineering, vol. **23, no. 4**, pp. 8-17, 1 July-Aug. 2021, doi: 10.1109/MCSE.2021.3085102.