Science & Technology Facilities Council

High Performance Computing driven software development for next-generation modelling of the world's oceans

Xiaohu Guo, Gerard Gorman, Mike Ashworth,
Stephan Kramer, Matthew Piggott, Andrew Sunderland.
ARC, CSE Department, STFC,
AMCG, Department of Earth Science and Engineering,
Imperial College London

CUG 2010

dCSE ICOM Collaborations

- Applied Modeling and Computation Group, Imperial College, London (AMCG, http://amcg.ese.ic.ac.uk/)
- ARC, The Computational Science & Engineering Department (CSED), STFC (http:// www.cse.clrc.ac.uk/)
- Proudman Oceanographic Laboratory, Liverpool (POL, http://www.pol.ac.uk/)

Science & Technology Facilities Council

INTRODUCTION

- Overview of Imperial College Ocean Model (ICOM) – the next generation ocean model
- Solver Comparison
- Profiling and Performance Analysis
- Summary

Science & Technology Facilities Council

Motivations for the next generation ocean model

- To resolve a wide range of spatial and temporal scales
- Model internal waves, boundary currents, eddies, overflows, convection events, …, accurately and efficiently within a global and coupled context
- Need for accurate and efficient representation of highly complex domains
- Ability to model interaction of flow with small scale topography, shelf seas, coastal regions, islands, estuaries, harbours,…

Science & Technology Facilities Council

A overview of the Computational Characteristic of ICOM

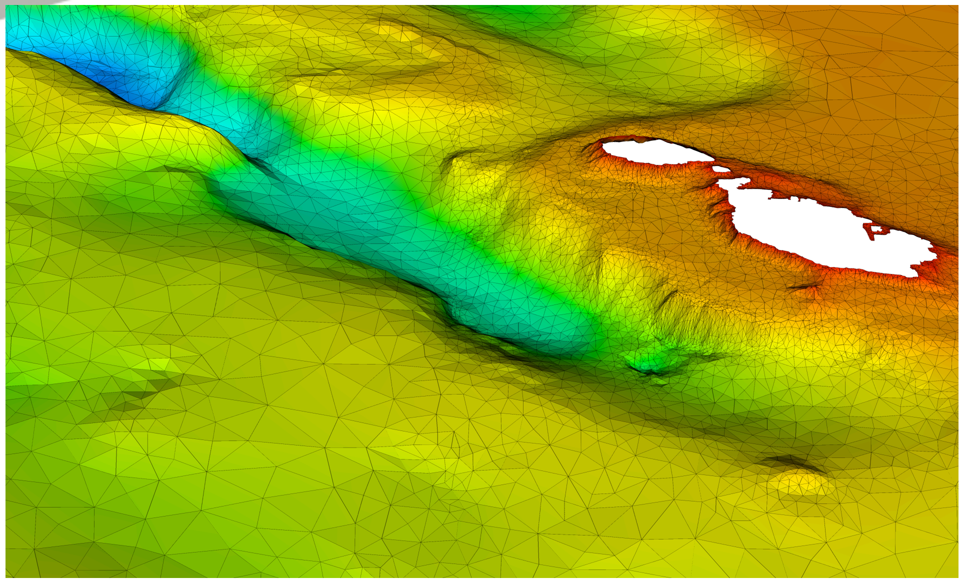
- Unstructured FEM Code
  - Start with Fluidity – an open source control volume finite element solver for 3D compressible multi-phase fluids. Has been developed by AMCG for more than a decade and is the basis for a range of multi-physics multi-scale applications
  - Initial mesh generation to follow complex bathymetry and coastlines -- **terrno**
- Adaptive Mesh, solving from large scales to small scales.
  - Add an adaptivity library which performs topological operations on the mesh, and mesh movement, to optimise the size and shape of elements in response to error measures
  - Dynamic load balance method -- **Zoltan**

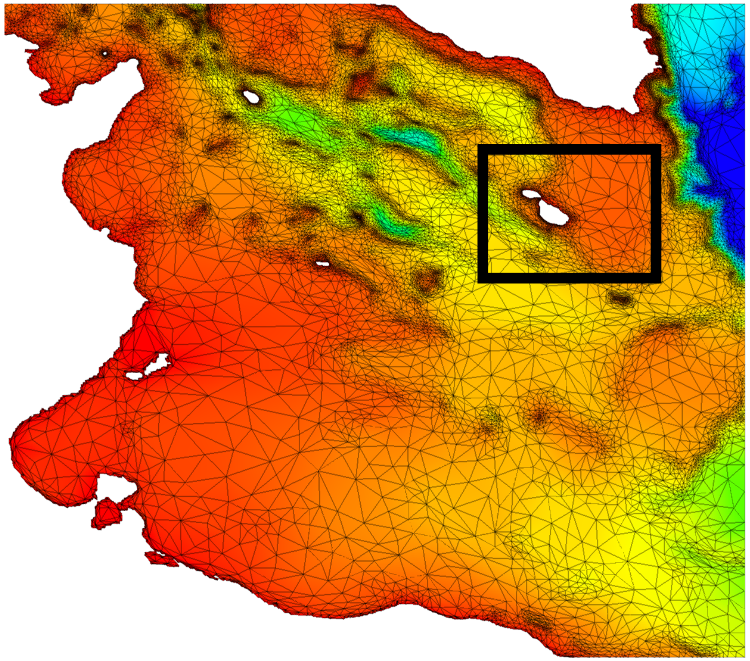
- Most time spent solving Ax=b, where A is a Sparse Matrix
  - FEM Matrix assembly
  - Using PETSc's preconditioner and Iterative Solver
  - Most Computing time is spent here
- Fortran, C++/C, Python MPI Based
- Makes use of open source solutions for I/O, Visualisation, etc
  - Advantage – using latest software features

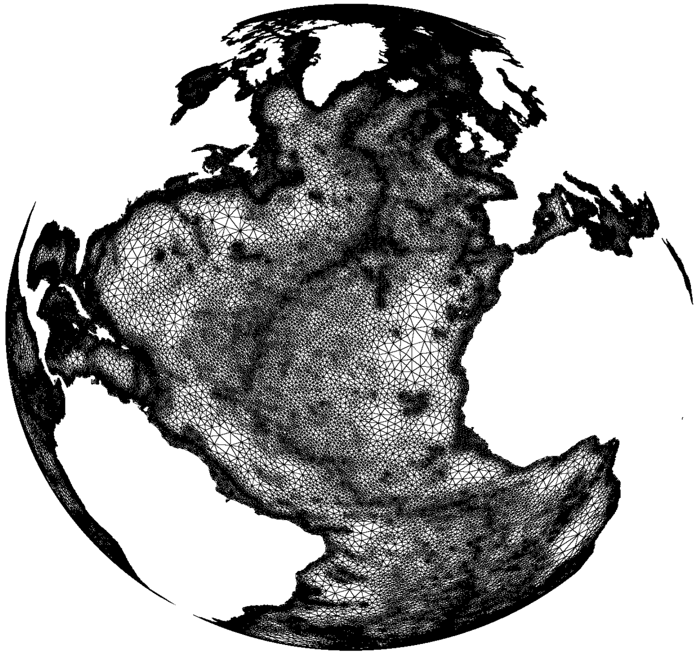
ICOM Software Package Lists

- VTK
- CGNS
- BLAS
- LAPACK
- XML2
- MPI
- PETSc
- ParMetis
- APPACK
- NetCDF
- UDUnits
- Python Development Environments
- Trang
- Spatial-Index
- Fortran 90 Compilers
- C++
- Subvision (SVN)

Science & Technology Facilities Council





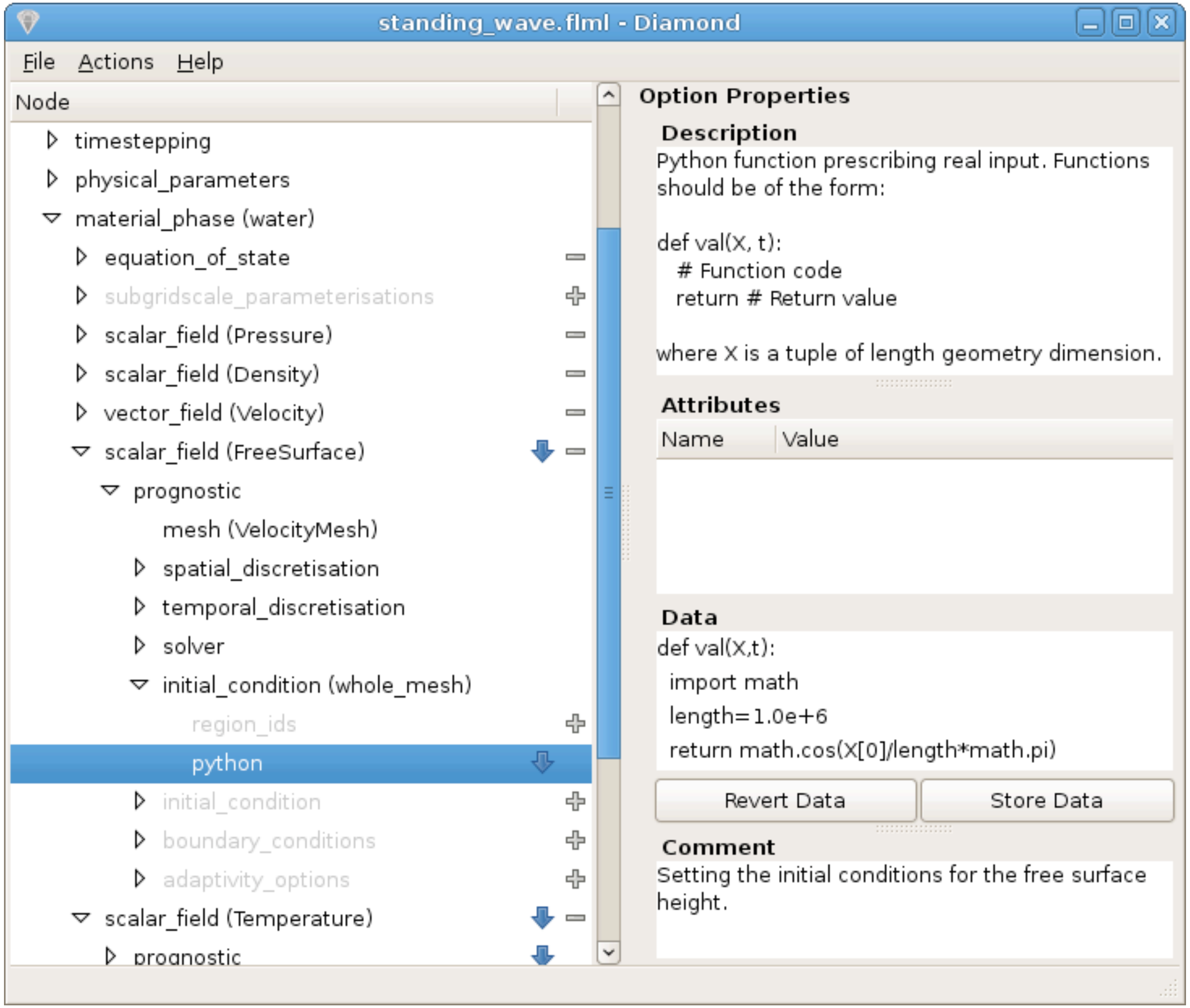


Unstructured meshes are an Ideal choice for representing complex problem domains and a coupled range of scales without the need for grid nesting

Science & Technology Facilities Council

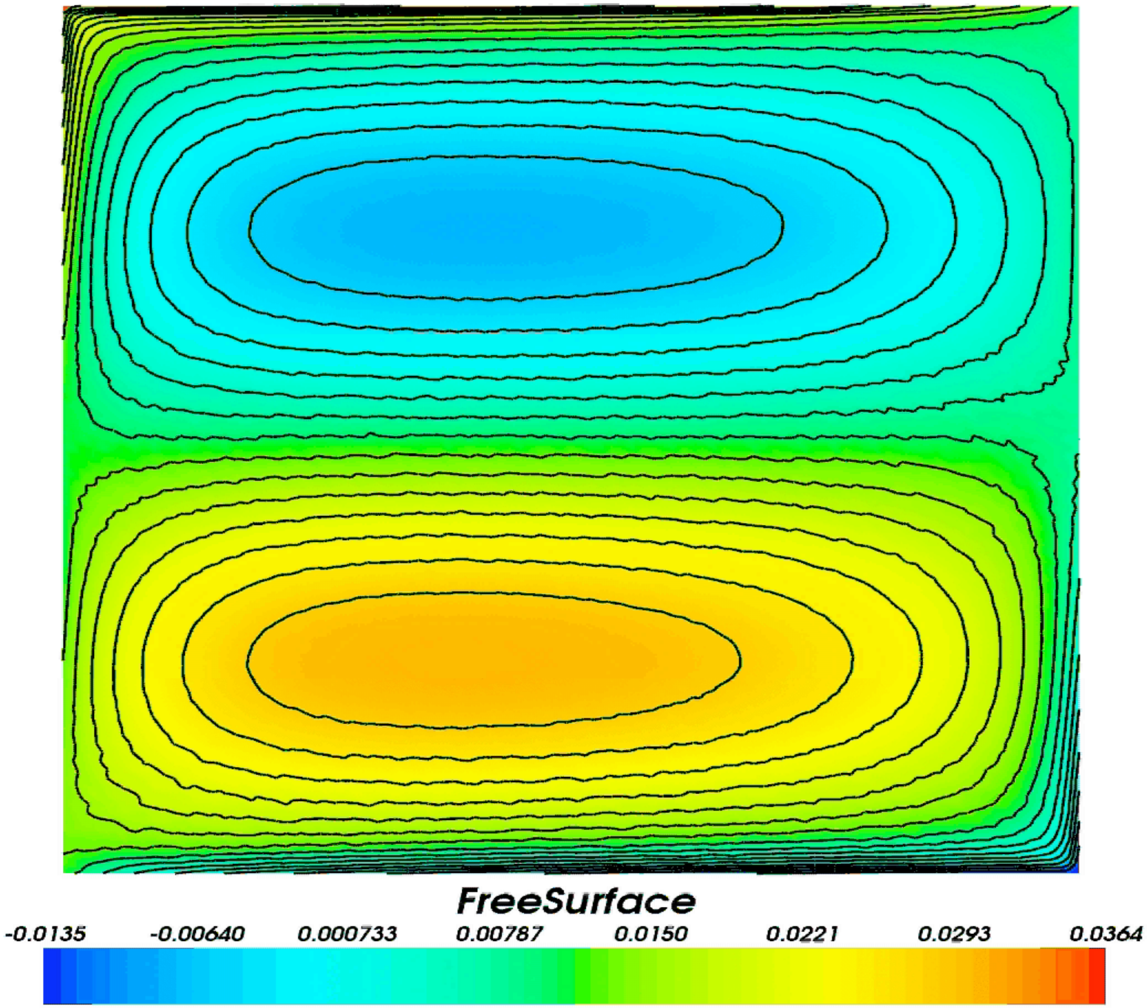
Diamond automatic pre-processing tool

- An xml schema file describes the rules that govern model options
- Diamond uses this to automatically generate a GUI based on the schema
- Options are entered and output as another xml file containing the options values
- This is read into an options library accessible from anywhere in code
- Includes many features, including the ability to define python functions executed at run time



Configuration of test case

- Baroclinic gyre benchmark test case has 10 million vertices; resulting in 200 million degrees of freedom for velocity
- The basic configuration is set-up to run for 4 time steps and not to adapt.
- Considering primarily the matrix assembly and linear solver stages of a model run.



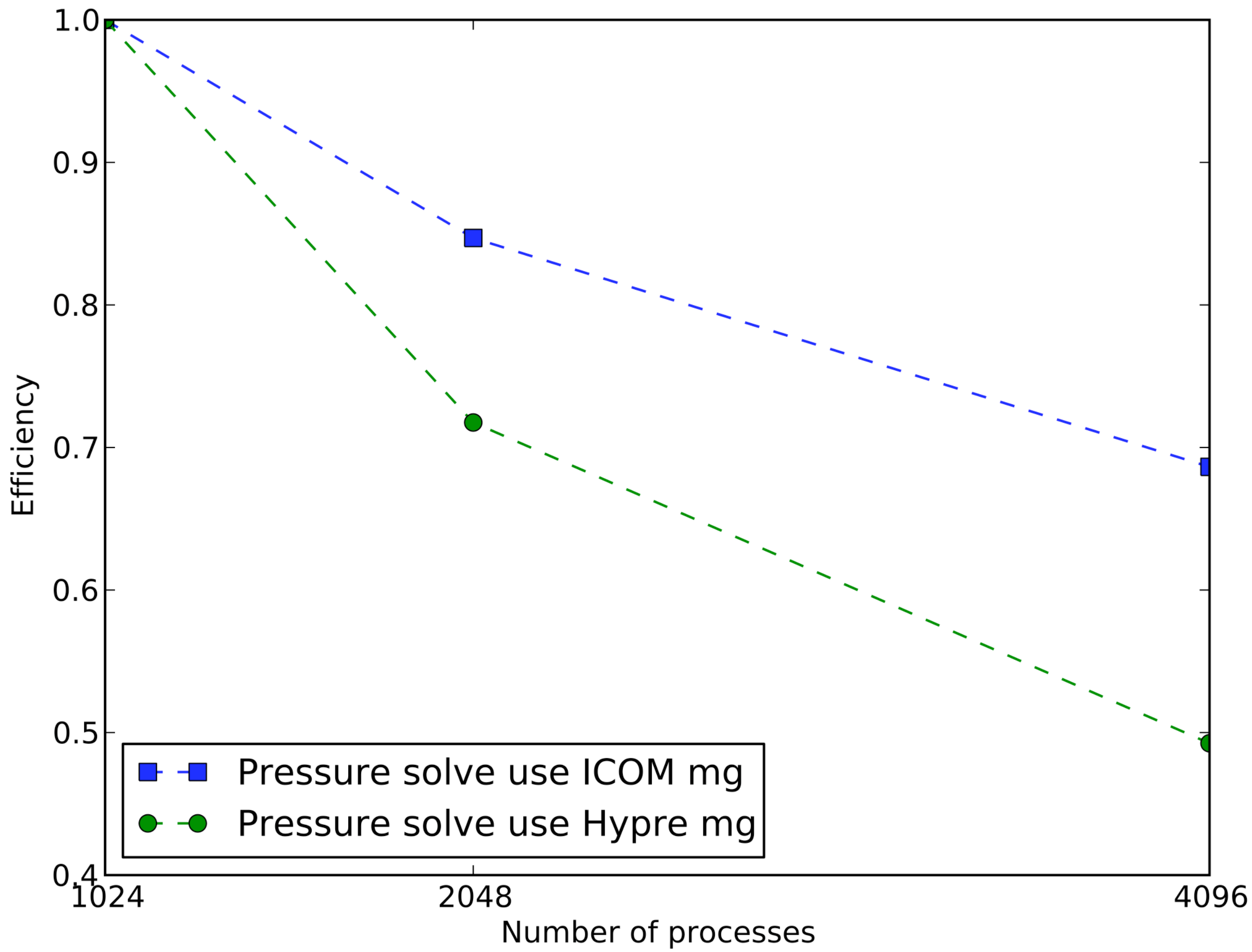
Science & Technology Facilities Council

Solver Comparisons

- The pressure matrix has a very high condition number

- ICOM MG targeted specially at large-scale, large aspect ratio ocean problems

- ICOM MG has better scalability than BoomerAMG due to its specialised nature.

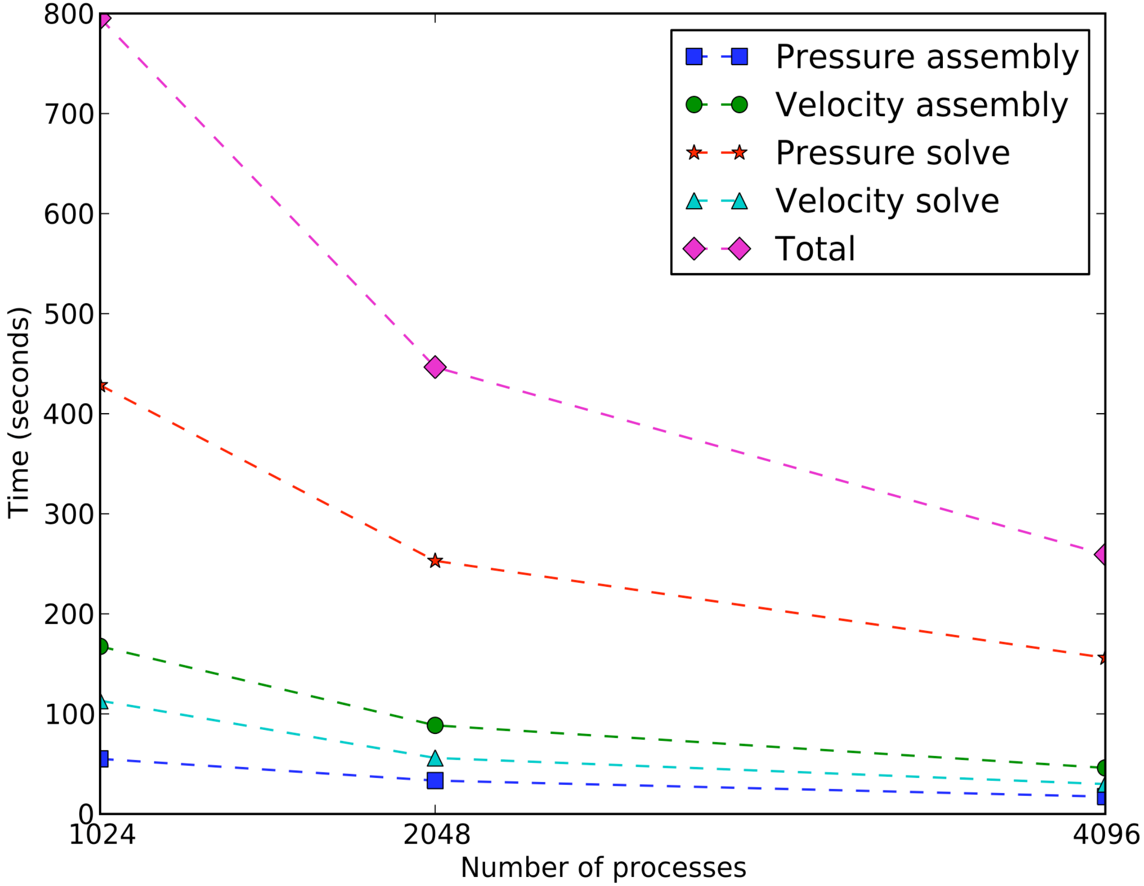


Profiling and Performance Analysis

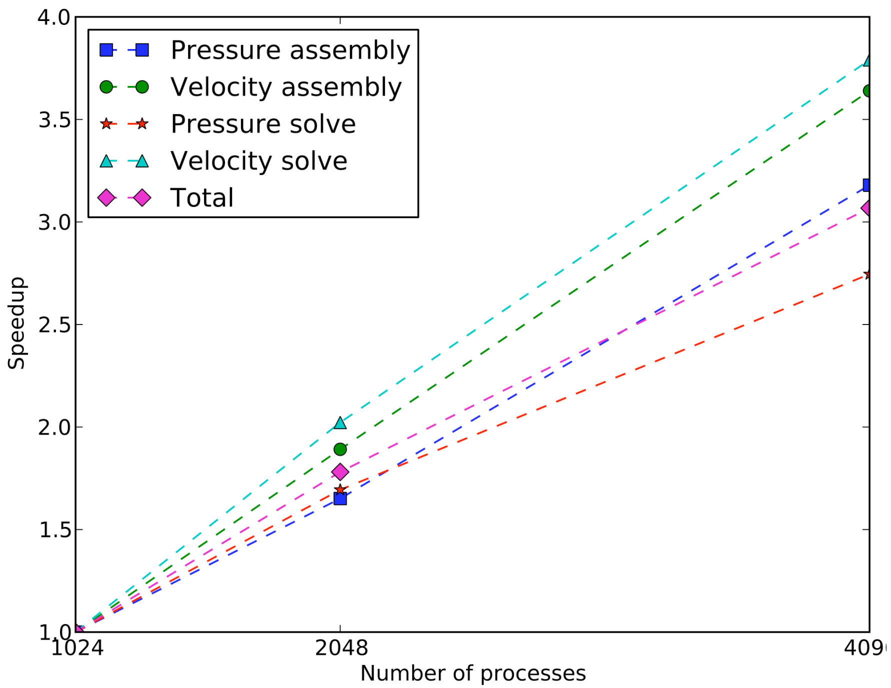
- Users should not spend time optimizing a code until after having determined where it spends the bulk of its time on realistically sized problems.
- Using CrayPAT/Vampir to address the parallel aspects, such as parallel efficiency, load balancing and communications overheads.
- Automatic tools in Profiling tools didn't work for ICOM profiling
- Simple timing hooks in the code to get a coarse grain profile of code performance

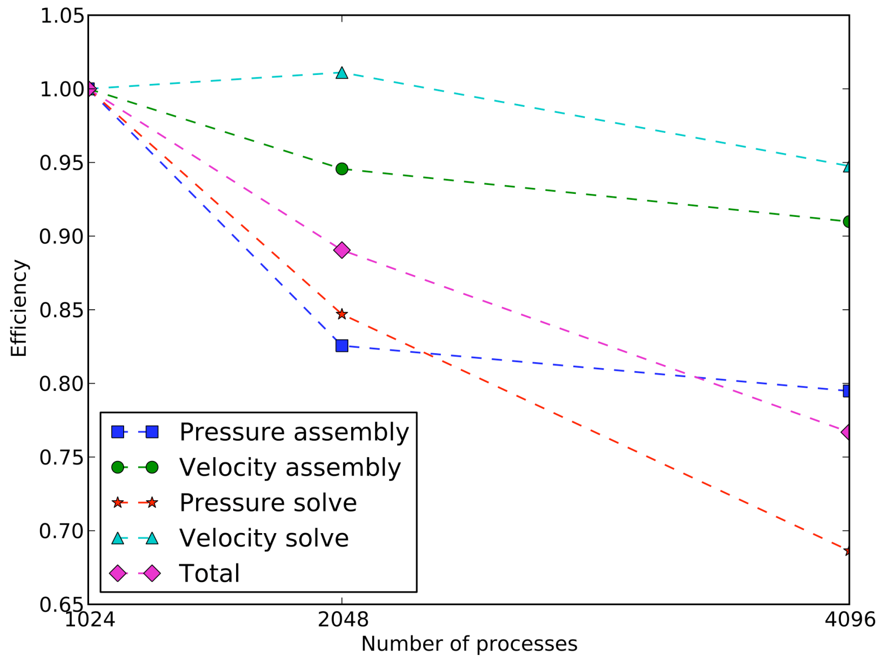
Basic Timings

- The solution process consists of the assembly of the linear systems representing the discretised momentum equation and the pressure equation.
- Matrix assembly for pressure and velocity can take more than 30% of the total simulation time with 1024 cores.
- Pressure solver is the main cost
- Matrix assembly phase is expensive
  - Significant loop nesting, where the innermost loop increases in size with increasing quadrature;
  - Indirect addressing (due to unstructured meshes)
  - Cache re-use.



Speedup and Efficiency



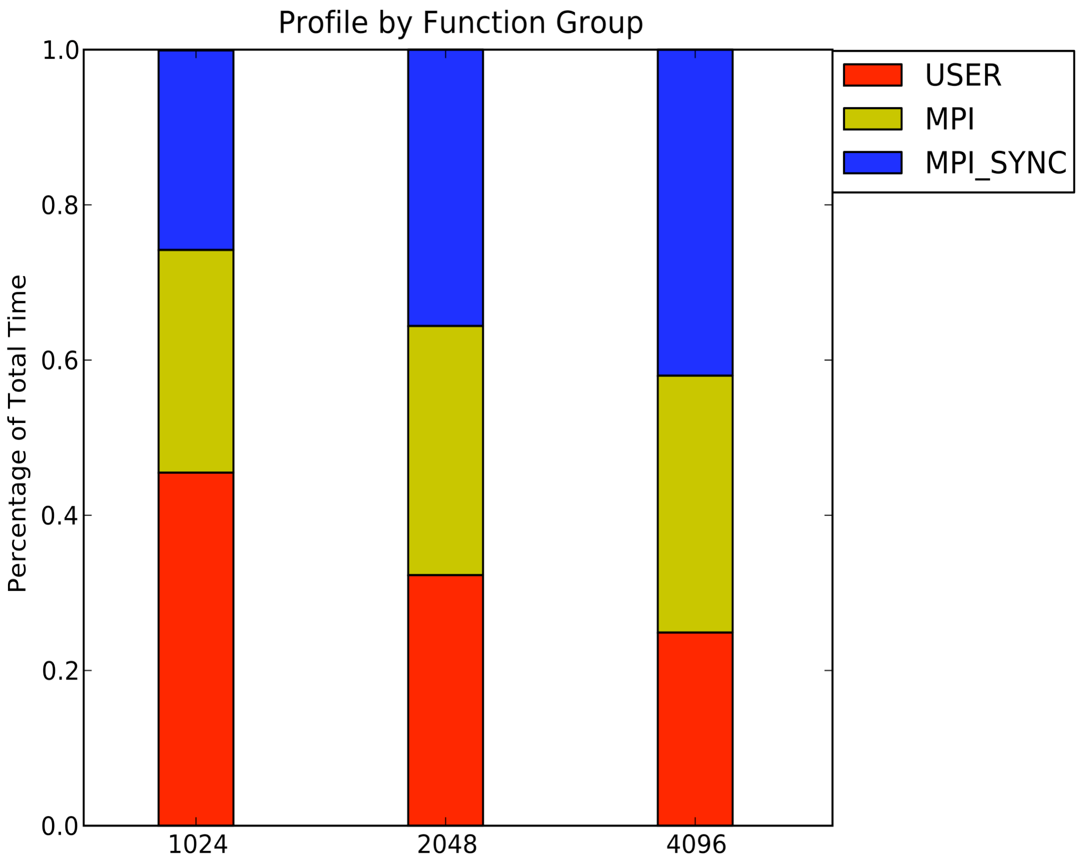


the speedup and efficiency of momentum solver and each of its components

Science & Technology Facilities Council

Communication overhead and load balance analysis

- Using **CrayPAT**, we obtained the statistic of three groups of functions, namely **MPI** functions, **USER** functions and **MPI_SYNC** functions.
- **MPI_SYNC** is used in the trace wrapper for each collective subroutine to measure the time spent waiting at the barrier call before entering the subroutine.
- The time percentage of MPI SYNC increases from 25.7% to 42.0%.
- The time percentage spent in MPI increases from 28.7% to 33.1% while USER functions drop from 45.5% to 24.9%

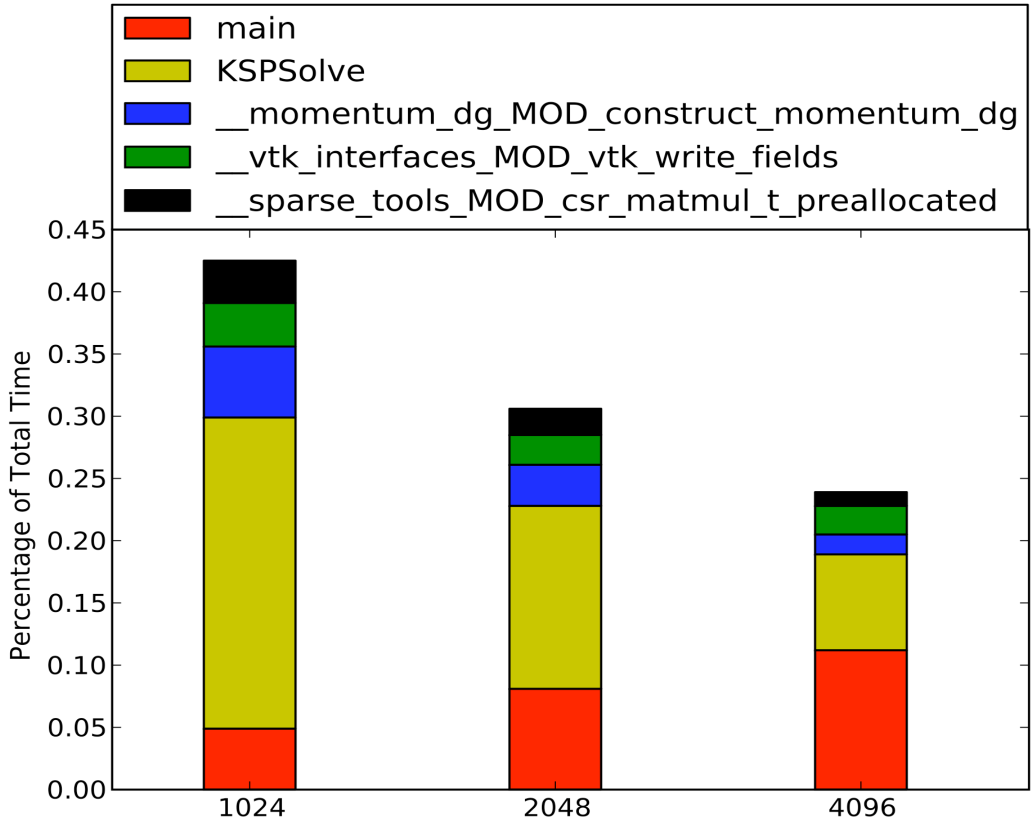


Top time consuming USER functions

- The speed up of the linear solver KSPSolve is about 3.5 with 4096 cores comparing with 1024 cores according to the CrayPAT tracing results.

- The function **main** represents the functions that have not been traced in the code. These functions are outside of momentum solver

- Future work will focus on these functions of poor scaling behaviour.

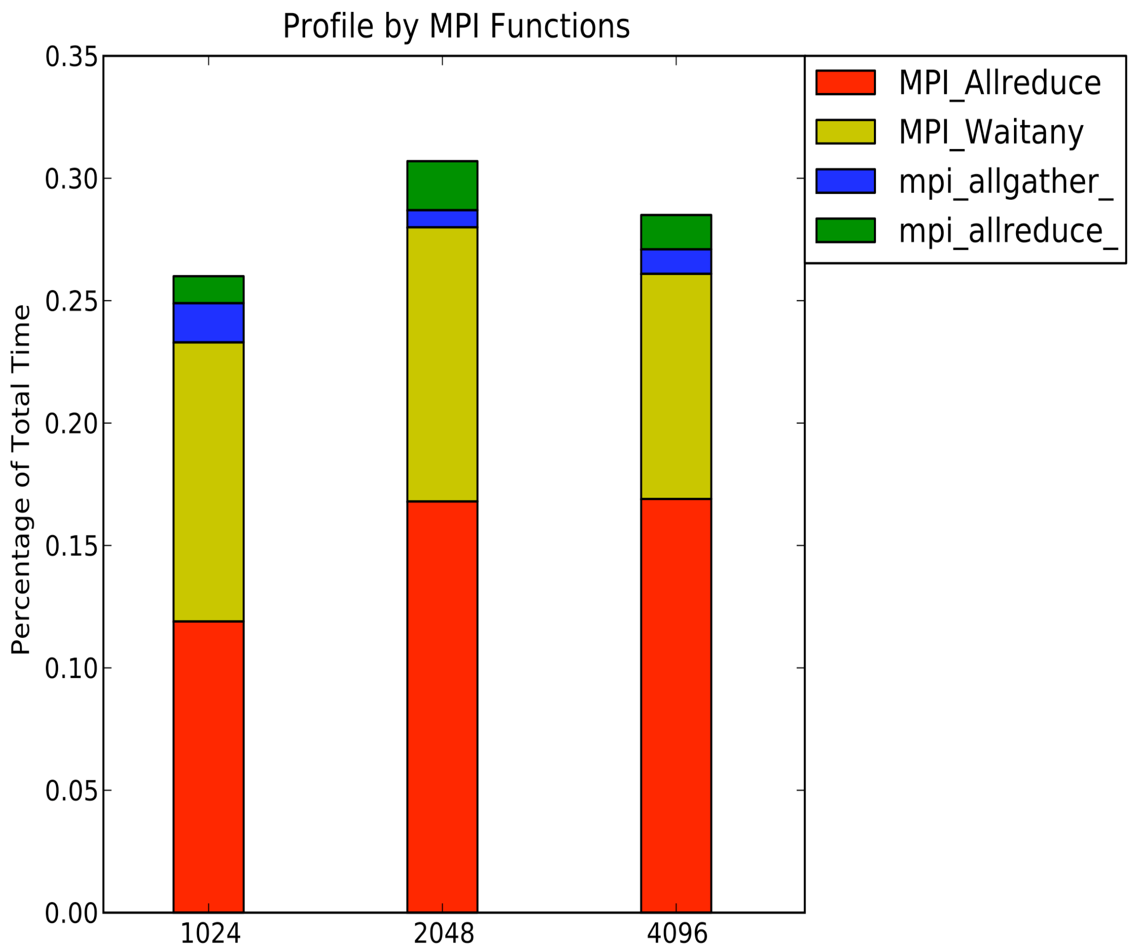


Top time consuming MPI functions

- The most time consuming of the **MPI** groups is **MPI_Allreduce.**

- From the call tree generated by **CrayPAT**, it becomes clear that this function is called from **PetscMaxSum** within **PETSc.**

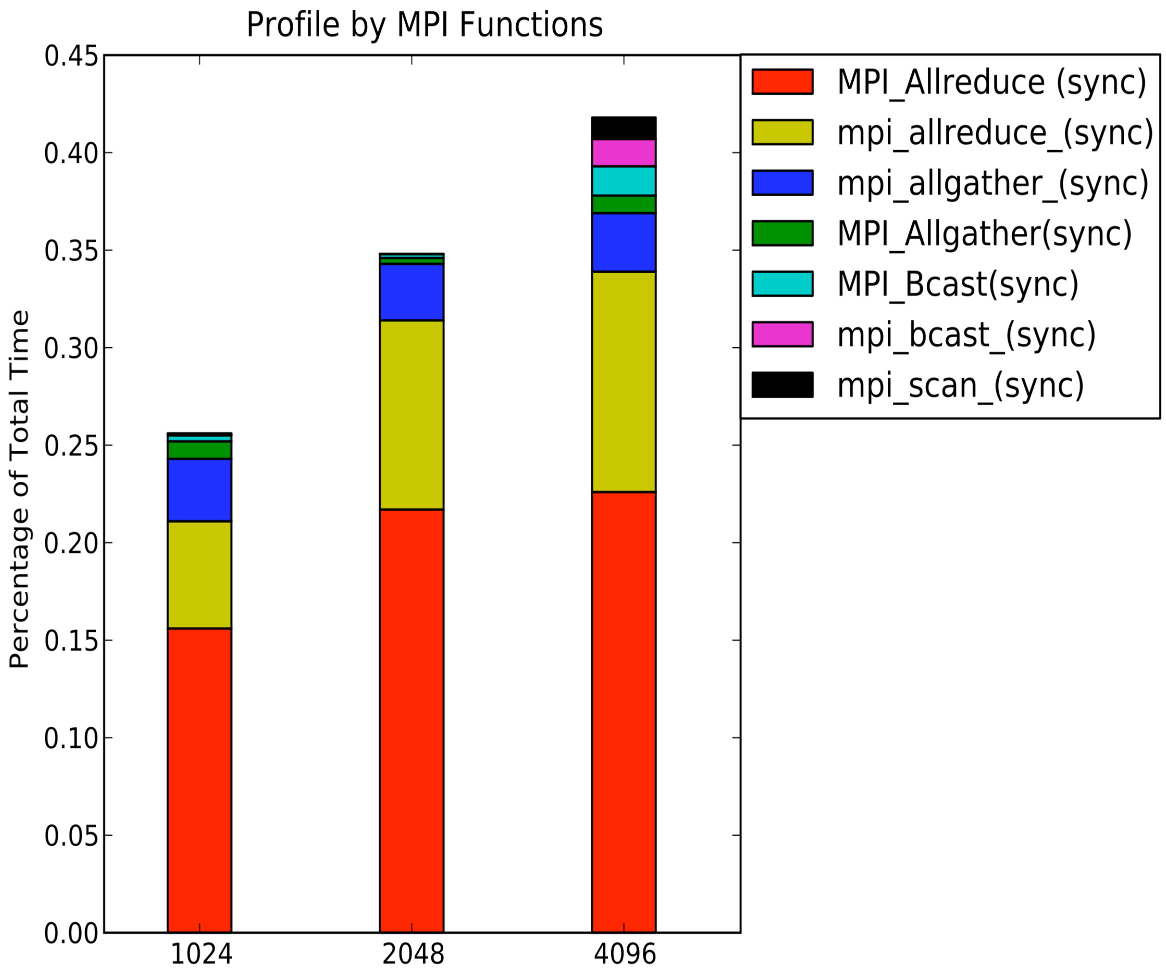
- **MPI_Waitany** is indicative of the quality of the load balancing. Given that this amount does not increase significantly between runs on 1024 to 4096 cores



Top time consuming MPI_SYNC functions

MPI_Allreduce accounts the most part of waiting time spent in the barrier, it is worth to check if there are possibility to combine several **MPI_Allreduces** together.

MPI_Bcast and **MPI_SCAN** are becoming more significant on 4096 cores, compared to runs on 1024 and 2048 cores



Science & Technology Facilities Council

Guidelines for third party library tracing for ICOM

- Requiring direct access to the source file or the object file, which limits the analysis of third party software performance, like **PETSc**.
- Properly reducing the profiling data determines qualities of profiling.
- Coarse time profiling + Fine grain profiling of specific parts of the code with CrayPAT/Vampir has been effective for ICOM

Summary

- From a starting point where the code was only routinely run on 64 cores on a local cluster, the **ICOM dCSE** project has significantly improved the performance of the code to enable efficient usage of large high performance computing systems such as the Hector Cray XT4.
- Presently the code is now scaling well up to at least 4096 cores on **HECToR**.
- Porting the code to **HECToR** has involved several challenges.
  - the code requires a range of third party libraries which need to be maintained on the target platform
  - Some Fortran 95 programming constructs caused compiler issues (stress-tested) for the various compilers. Resolving these required substantial effort from different groups including the developers, STFC ARC group and HECToR Support.
- Profiling the real world applications is a big challenge
  - Need to reduce the profiling data size whilst maintaining a representative dataset
  - Manual instrumentation was required in order to focus on specific sections of the ICOM code.
  - **CrayPAT** and **Vampir** are well suited to fine grain profiling on specific sections of the code

Science & Technology Facilities Council

Acknowledgements

- The authors would like to acknowledge the support of a **HECToR** *distributed Computational Science and Engineering* award.
- The authors would also like to thank the **HECToR and NAG support team** for their help throughout this work.
- Gerard Gorman gratefully acknowledges support from the **Leverhulme Trust**.
- Some experiments of this paper has been carried on the **Swiss National Supercomputing Centre's Cray XT5, Rosa,** and we would also like to thank their support team.

THANKS !