Hybrid CPU/GPU FE$^2$ multi-scale implementation coupling Alya and Micropp

Guido Giuntoli∗
guido.giuntoli@bsc.es
Barcelona Supercomputing Center
Barcelona, Spain

Judicaël Grasset
judicael.grasset@stfc.ac.uk
STFC, Daresbury Laboratory
Warrington, United Kingdom

Alejandro Figueroa
afiguer7@masonlive.gmu.edu
George Mason University
Washington, United States

Charles Moulinec
charles.moulinec@stfc.ac.uk
STFC, Daresbury Laboratory
Warrington, United Kingdom

Mariano Vázquez
mariano.vazquez@bsc.es
Barcelona Supercomputing Center
Barcelona, Spain

Guillaume Houzeaux
guillaume.houzeaux@bsc.es
Barcelona Supercomputing Center
Barcelona, Spain

Stephen Longshaw
stephen.longshaw@stfc.ac.uk
STFC, Daresbury Laboratory
Warrington, United Kingdom

Sergio Oller
sergio.oller@upc.edu
Universitat Politécnica de Catalunya
Barcelona, Spain

ABSTRACT
This poster exposes the results of a new implementation of the FE$^2$ multi-scale algorithm that is achieved by coupling the multi-physics and massively parallel code Alya with the GPU-based code Micropp. The coupled code is mainly designed to solve large scale and realistic composite material problems for the aircraft industry. Alya is responsible of solving the macro-scale equations and Micropp for solving the representation of fibres at the microscopic level. The poster shows computational performance results that demonstrate that the technique is scalable for real size industrial problems and also how the execution time is dramatically reduced using GPU-based clusters.

CCS CONCEPTS
• Computing methodologies → Parallel programming languages;
• Applied computing → Physics.

KEYWORDS
Multi-Scale, GPU, OpenACC, Code Coupling

1 INTRODUCTION
The main algorithm analysed in this work is the Finite Element Squared (FE$^2$) multi-scale method [1]. This is generally used to solve highly heterogeneous composite material problems, for example, a carbon fibre panel of an aircraft. The implementation that is proposed couples the Finite Element (FE) multi-physics code Alya [2] to solve the macro-scale equations and the FE code Micropp [3], developed from scratch at the BSC, for the micro-scale homogenisation. The latter has been ported to GPUs to solve the microstructure. This is significant as this calculation is typically the most computationally intensive of the coupled problem and so accelerating using GPUs enables larger and higher quality cases to be solved.

The coupling is used for simulating the compression of a composite material panel belonging to the fuselage of an aircraft (see Fig. 1). In the FE$^2$ multi-scale algorithm, Alya transfers the macro-scale deformations to Micropp [4]. This then uses the macro-scale deformations for setting the boundary conditions on a Representative Volume Element (RVE) of the microstructure which then calculates the average macro-stress by solving an FE problem locally. Finally, Micropp passes the macro-stress back to Alya and the coupled process is repeated until a steady state is achieved. This is the basic methodology but different complex coupling schemes such as a one-way or full coupling schemes can be applied [5]. For this work, the simulations were considering the one-way scheme and non-linear problems for modelling damage, these being two representative assumptions for industrial applications.

2 COMPUTATIONAL IMPLEMENTATION
The macro-scale code, Alya is an MPI-based code for solving multi-physics applications, that can also support OpenMP and OpenACC.
It is designed to solve large FE problems up to billion degrees of freedom on the newest architectures. It is one of the 12-only software of the Unified European Applications Benchmark Suite of PRACE [2] and has shown a good weak scalability up to 100K cores on the Blue Water (Cray XE6) supercomputer [2]. It is used to solve the macro-scale solid mechanics part of the FE\(^2\) multi-scale algorithm, and MPI serves to couple it to the Micrpp code to create the full FE\(^2\) framework. Because the macro-scale is much less computationally demanding than the micro-scale, the macro-scale code is used with MPI only.

The micro-scale code Micrpp was designed from scratch to efficiently solve the FE problem inside a RVE. A transport equation for a stress tensor is solved implicitly in time on a domain using a structured grid, ensuring direct addressing. The code is written in C++ and supports coupling with other software written in C and/or Fortran as for instance Alya (Fortran 95). It was ported to GPU using OpenACC focusing on the three most costly parts of the code, e.g. matrix assembly, the right-hand side of the tensor equation, and the conjugate gradient algorithm for the transport equation.

The combination of a macro-scale MPI-based code with a CPU/GPU accelerated micro-scale code makes running the coupled solution on distributed clusters with multiple GPU devices possible and affordable. The implementation also allows for GPU overloading, i.e. several micro-scale problems are solved concurrently on the same GPU. This increases the occupancy rate of the GPU and the parallel efficiency of the code.

A panel as shown in Fig. 1 is made of several different materials, with some parts only made of fibres, a priori identified when setting up a simulation. Simulating the micro-scale is the most computationally expensive part of the coupling, leading to very different loads for the nodes, depending whether they solve the micro or the macro-scale. To circumvent load imbalance, a weighting is applied during the partitioning at the macro-scale stage, using Metis [6].

The poster indicates clearly the percentages of the time spent in each of these three functions of the micro-scale algorithm for a pure CPU execution and the hybrid CPU/GPU version. The test case presented is accessible to the public in order to be reproduced and constitutes a benchmark of Micrpp [7].

3 RESULTS

All the simulations and results shown were performed on the CTE-POWER cluster [8]. The cluster is comprised of 50 computing nodes (and 2 login nodes) with 2 IBM Power9 8335-GTH (40 cores per node) and 4 Nvidia V100 GPUs, with 16GB of memory each. The simulations were carried out on up to 32 nodes giving a total of 1280 CPU cores and 128 V100 GPUs. In all cases the macro-scale mesh is composed of 140K elements with 9K elements for which the coupling is active, as they are in the composite material region. For each of these 9K macro-scale integration points a micro-scale problem is solved.

The poster shows a comparison between the calculation time using CPU only and CPU/GPU. The test case is run for different micro-scale mesh resolutions of 25\(^3\), 50\(^3\), 75\(^3\) and 100\(^3\) elements. Being the simulation almost wholly controlled by the micro-scale part, the results show that for micro-resolutions larger than 25\(^3\) elements the implementation with CPU/GPU is much superior in a calculation time term than with CPU only. For the 100\(^3\) case, the speedup achieved with the CPU/GPU approach is of ×3.9, showing the benefit of porting the coupling to GPUs.

The last result of the poster, consisting in a non-linear simulation of 20 time steps and a micro-resolution of 50\(^3\) elements, shows a parallel efficiency of 91% from 8 to 16 nodes and 81% from 8 to 32 computing nodes. It demonstrates that the algorithm can be used in real composite problems for industrial purposes and that the efficiency achieved with the hybrid CPU/GPU is much better than with CPU only. In Fig. 2 the actual improvements of this work, in macro- and micro-scale size terms, are compared to other implementations in the literature.

![Figure 2: The current magnitude of problems that can be solved with the FE\(^2\) multi-scale implementation. Comparison with other works.](image)

ACKNOWLEDGMENTS

The authors would like to thank to the Barcelona Supercomputing Center (BSC) for the computational resources provided, to the 11\(^{th}\) and 12\(^{th}\) Severo Ochoa Mobility Grants (BSC) and to the Energy Oriented Centre of Excellence II (EoCoE-II), grant agreement number 824158.

REFERENCES