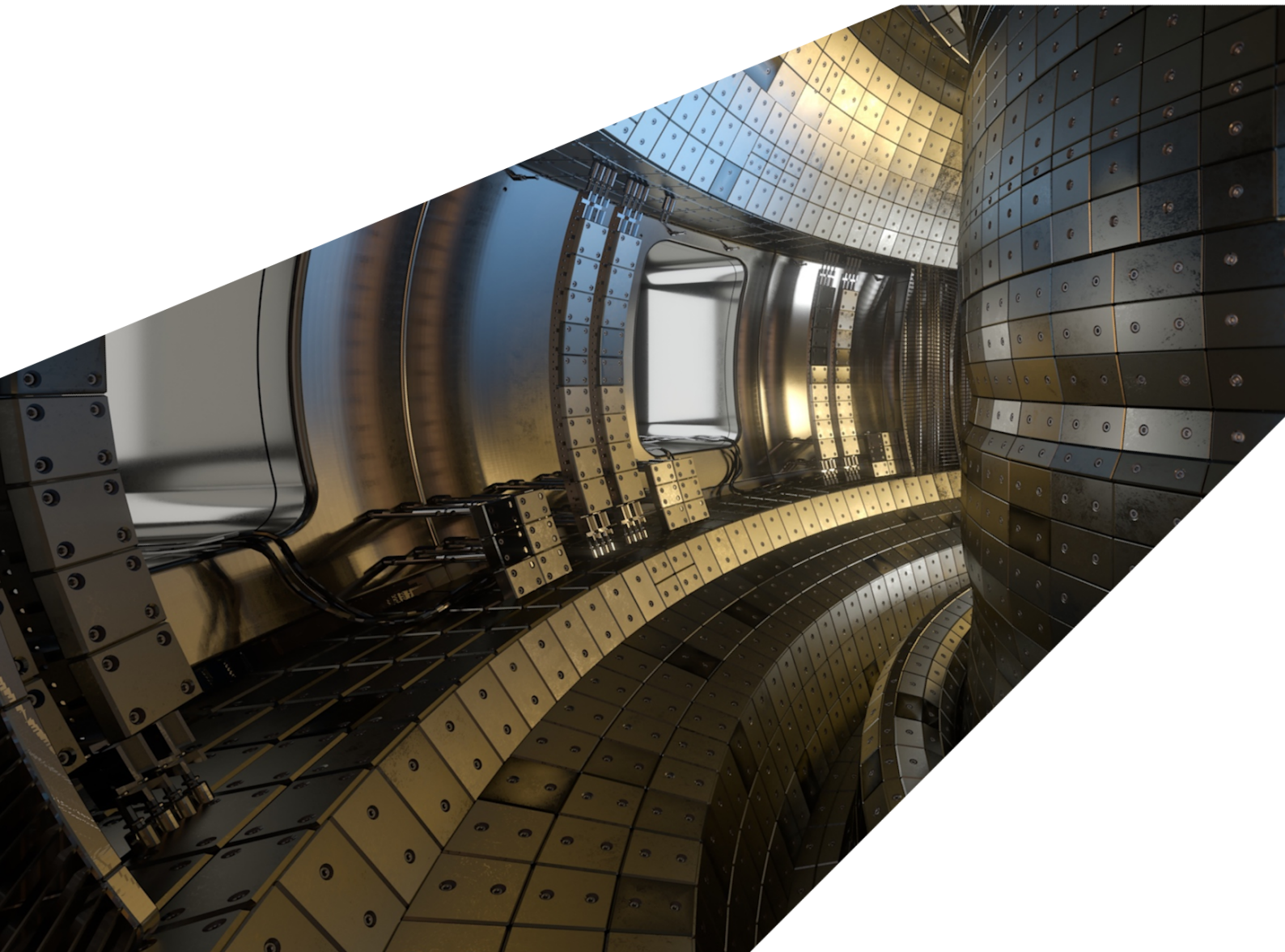
UK Atomic Energy Authority

ExCALIBUR

High-dimensional Models Complementary Actions 2

M4.3 Version 1.00

Abstract
The report describes work for ExCALIBUR project NEPTUNE at Milestone M4.3. Part involves a 2-D model of neutral gas and impurities with critical physics and is relevant to work packages FM-WP2 and FM-WP3. We provide an overview of the inter-species interactions that are expected to occur in the SOL. We describe an extension to a particle implementation that allows global communication of particle data within in the simulation. We also describe results from 2-D ($1d1v$) experiments with Particle-in-Cell algorithms.



UKAEA REFERENCE AND APPROVAL SHEET

	Client Reference:	
	UKAEA Reference:	CD/EXCALIBUR-FMS/0062
	Issue:	1.00
	Date:	18 March 2022
Project Name: ExCALIBUR Fusion Modelling System		

	Name and Department	Signature	Date
Prepared By:	Will Saunders James Cook Wayne Arter BD	N/A N/A N/A	18 March 2022 18 March 2022 18 March 2022
Reviewed By:	Wayne Arter Project Technical Lead	W. Arter	18 March 2022

1 Introduction

1.1 Critical Physics

A SOL simulation contains multiple different species with different physical properties. These properties may be constant such as mass or varying such as degree of ionisation. The behaviour of these species can be substantially different depending on the particular physical environment these species are exposed to. For example a particular species may travel with such high velocities that techniques such as time averaging must be utilised to capture and represent very short time scale effects without resorting to severe time step constraints. Other species may move relatively slowly but with a high rate of inter-particle collisions. These collisions are expensive to model exactly but, depending on the corresponding distribution functions, may be approximated by a fluid representation.

Multiple sources and sinks exist for each of the constituent species in a SOL simulation. The plasma itself may consist of Deuterium and Tritium ions, and the corresponding electrons, that form the fuel for the fusion reaction. The exhaust from a successful fusion interaction is a helium ion, alternatively known as an Alpha particle, which is accompanied by a free neutron. These fuel species are accompanied by ionised neutrals and molecules that exist in the plasma intentionally and through impurities. Neutral species are introduced into the reaction vessel for the purposes of heating, cooling, and diagnostics. Neutrals also impact with plasma facing surfaces. Furthermore, plasma interactions with the vessel wall emit impurities into the reaction vessel known as “sputter”.

These species do not exist in isolation but continuously interact with each other throughout the simulation. These inter-species interactions are responsible for the transfer of mass and energy between the constituent species that form the simulation and should be accurately modelled such that the simulation is representative of the underlying physical processes. A list of example inter-species interactions typically found in the SOL is as follows:

Ionisation
: A neutral species looses, or potentially gains, one or more electrons due to an interaction with the plasma. The motion of this, now charged, species is influenced by the electric and magnetic fields. In a simple model constructed with only plasma and neutral species, ionisation is a source of plasma species and a sink of neutral species that together conserve mass. Removing an electron from a species requires energy to extract the electron from the electrostatic potential well of the host nucleus.

Recombination
: In cooler regions of the plasma electrons may not carry enough energy to escape the electrostatic potential well nearby to an ionic species. The free electron recombines with the ionised species to form a neutral atom which is no longer influenced by the magnetic and electrostatic fields. As the electron reattaches to the ion by falling down a potential well energy is released as radiation.

Charge Exchange
: Charge exchange is a process where a neutral species and ionic plasma species interact. For example a fast moving ion could collide with a slower moving neutral. We could assume

that in the collision an electron is transferred from the neutral to the plasma species without a loss in energy. If we further assume that the ion and neutral have nucleus of equal mass then this charge exchange process can be considered as a transfer of momentum from plasma species to neutral species. In highly collisional regions of the simulation these charge exchange interactions are computationally expensive to model as particle species and may be adequately approximated by a fluid representation as discussed by Borodin et al. [1].

Excitation

An electron attached to a neutral species may be excited into a higher energy state through neutral-plasma interaction. This excitation requires energy which is provided by the plasma and hence cools the plasma. When the electron falls returns to a lower energy state the energy equal to the difference in electrostatic potential is emitted as radiation.

Radiation

Electromagnetic radiation occurs when a charge undergoes acceleration and this continuously occurs in a magnetically confined plasma. The acceleration of plasma species due to magnetic fields emits radiation known as cyclotron radiation. Furthermore, Coulomb interactions between charged species emit bremsstrahlung radiation. Impurities cause enhanced bremsstrahlung radiation due to their higher ionic charge states in comparison to fuel species. A more detailed description of radiation in plasmas is presented by Wesson et al. in [2].

Correct modelling of these inter-species processes is essential to capture and represent the correct physics present in the plasma. The exact implementation of the interactions depends on the particular representation and method of evolution used for each species. For example highly collisional species may be represented by a fluid approximation with time evolution described by a set of coupled equations for mass, momentum and energy. In this fluid case the interactions exist as source and sink terms in the corresponding equations. Alternatively a particle based kinetic description may use inter-particle interactions to compute these processes.

In the next section of this report we discuss methods to efficiently implement the bookkeeping operations of fast moving particles, such as neutrals, in a distributed memory computation. In the final section of this report we describe results of 2-D $(1d1v)$ Particle-in-Cell experiments

2 Neutral Particle Transport

2.1 Introduction

In particle simulations such as Molecular Dynamics (MD) it is common to assume that individual particles move relatively slowly and that the mean flow is zero in the sense that particle motion is normally distributed with zero mean flow in any dimension. MD codes that implement a domain decomposition approach and assume that when a particle leaves a region of space owned by an MPI rank that the destination rank is a neighbour of the MPI rank which owns the region of space the particle is departing. This assumption of a local transfer of ownership permits a communication pattern which is also local, ie. MPI ranks only have to exchange send and receive particle counts and data with adjacent MPI ranks as opposed to all other MPI ranks.

In contrast to MD simulations of relatively cool bulk material, in particle simulations of plasma, such as PIC, there may be considerable bulk flow of a particle species parallel to the magnetic field lines and considerably less flow perpendicular to the magnetic field lines. This anisotropy in the underlying physics will result in anisotropy of the communication pattern that transfers particles between MPI ranks. In addition, neutral species are not confined by the magnetic field lines and are free to propagate through the simulation domain until colliding with the domain geometry or a physically relevant process occurs such as a collision with another particle.

The velocities of a neutral species may be highly directional such as in diagnostic cases where a species of neutrals is injected into the plasma. Alternatively in regions of the simulation domain with higher collision rates the neutral species may exhibit a much more isotropic and diffusive velocity distribution. We also expect cases where neutral particles carry a very significant velocity that allows an individual particle to cross the simulation domain in a small number of time steps. As the neutral carries no charge and is unaffected by the magnetic field it is not known a priori which direction this particle will travel.

If the MD approach we describe is implemented then the velocity of a particle would be limited to the size of the subdomains owned by individual MPI ranks. This restriction would artificially truncate the velocity distribution of a fast moving species which is undesirable and an implementation induced limitation. Instead we investigate efficient mechanisms to allow the transfer of particle ownership even in the case of fast moving particles.

In the remainder of this section we describe a two stage approach to transfer the ownership and data of a particle between MPI ranks. In the first stage the new owning rank of a departing particle is identified by inspection of the particle position. The second stage is a bookkeeping operation that transfers the data of departing particles to the new owning ranks and receives the data for the incoming particles.

2.2 Ownership Identification

We describe an approach that uses both the unstructured mesh provided by the finite element library and an additional regular Cartesian mesh which is imposed over the entire domain. In Figure 1 the FEM mesh is the unstructured mesh drawn in solid black and a single cell of the

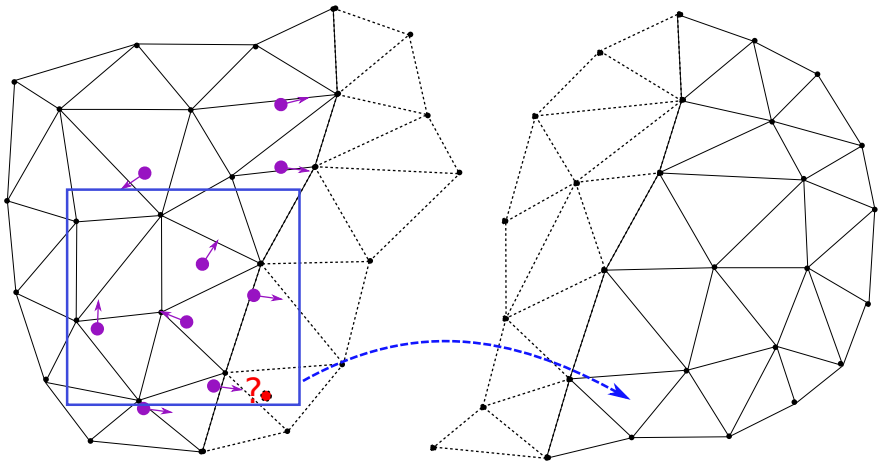


Figure 1: Illustration of locally owned unstructured mesh (solid black) extended with a halo region based on physical width (dashed black). Imposed Cartesian mesh indicated by a single cell (solid blue). Purple dots and arrows represent particles and direction of motion respectively.

imposed Cartesian mesh is drawn in solid blue. The Cartesian mesh provides a simple method to map a position in space to an MPI rank that either owns the destination location of a departing particle or has the destination location in the halo region of the FEM mesh on that MPI rank. In this report we proceed on the basis that a Cartesian grid structure is suitable to demonstrate and assess this approach.

A more complex mesh, such as one that conforms to magnetic field structure or resolves small features, may consist of cells which drastically vary in size. In this more complex scenario a single regular Cartesian mesh would be unlikely to provide suitably sized cells over the whole domain. Instead a multi-level approach such as an Octree style structure would allow adaptive refinement in regions where a single large cell would impose unreasonably large halos on each MPI rank.

Typically the parallel decomposition approaches of meshes for FEM only duplicate the mesh entities which are shared between MPI ranks in the spatial decomposition. These mesh entities are usually restricted to entities which hold degrees of freedom which are shared between multiple cells as the parallel decomposition is performed along edges and vertices. Hence entire mesh cells are not typically shared between adjacent MPI ranks. Furthermore simulation techniques that extend the local representation of a mesh with a halo region typically add a halo region that is constructed with a known fixed width given in terms of mesh connectivity. For example in a classic finite difference formulation on a Cartesian mesh with a centred 3-point stencil only a single neighbouring mesh cell is required. In the approach we describe the stencil must be large enough to at least cover the region of the domain that is assigned to each MPI rank by the imposed Cartesian mesh, ie. the area under the blue square in Figure 1.

Algorithm 1 describes the process that determines the destination rank of a departing particle. The algorithm assumes that for each cell in the halo region the remote MPI rank that owns the halo cell is known. In Section 2.3 we discuss how the halo regions facilitate a local communication pattern between MPI ranks which own spatially close regions of the simulation domain. This local communication pattern is used for departing particles where the destination rank is known exactly

by inspection of the halo cells. Particles which do not travel to a MPI rank determined by the halo cells are considered to have global movement and are transferred via a global communication pattern.

Algorithm 1 Determine destination ranks of departing particles through a global transfer mechanism and halo regions.

Require: Global map from position to MPI rank using Cartesian grid G.
Require: Local map from position to MPI rank using halo cells H.
  for departing particle i **do**
    Let new owning rank $R_i = -1$ (undefined)
    Let $\vec{r}_i$ be the new position of particle i
    if $\vec{r}_i$ in halo **then**
      $R_i = H(\vec{r}_i)$
      Mark i as a local move
    else
      $R_i = G(\vec{r}_i)$
      Mark i as a global move
    end if
  end for
  Transfer particles to remote ranks using global communication pattern as described in Section 2.3
  for newly received particle i from global transfer **do**
    if i in halo **then**
      Let new owning rank $R_i = -1$ (undefined)
      Let $\vec{r}_i$ be the new position of particle i
      $R_i = H(\vec{r}_i)$
      Mark i as a local move
    end if
  end for
  Transfer particles to remote ranks using local communication pattern as described in Section 2.3

As it is desirable to limit global communication in a distributed memory computation we consider the depth of the halo region to be a tunable parameter. From a statistical standpoint a wider halo region enables a larger number of particles to be transferred via the local communication pattern which we expect to be advantageous in regions where there is a significant directional component to the bulk flow. However regions with a lower particle density may benefit from a more narrow halo to reduce unnecessary communication between MPI ranks.

2.3 Transfer of Particle Data

The global nature of the physical process of the fast moving particles motivates a requirement for a global communication pattern. However global communication patterns are typically undesirable in distributed computing due to the overhead of the communication and implicit or explicit synchronisation that often occurs to ensure correctness. As discussed in the introductory Section 2.1 we

expect that a relatively small proportion of particles require a "anywhere to anywhere" communication pattern which we refer to as "global". For the implementation discussed in this report we utilise the MPI one-sided communication functionality which allows the programmer to define a so called access epoch in which a MPI rank can send data to a remote rank without a matching receive call on the remote rank. In our implementation this one-sided communication occurs between a pair of barriers that provide the synchronisation and our implementation aims to minimise the time spent between these barriers. An overview of this global communication approach is presented in Algorithm 2 and an overview of the neighbour based communication approach is described in Algorithm 3.

Algorithm 2 Overview of global move bookkeeping operation using one-sided MPI.

Require: On all ranks: MPI Window around buffer $C \in \mathbb{N}^0$ to count the number of remote MPI ranks that will send to this rank.
Require: List of remote MPI ranks S that this rank will send particles to.
Let C = 0
MPI Barrier on C communicator
for remote rank $s \in S$ **do**
Start access epoch for window C on rank s
Perform $C = C + 1$ atomically on s
End access epoch for window C on rank s
end for
MPI Barrier on C communicator
Simultaneously receive data from the C unique remote MPI ranks that hold particles to send to this rank and send data to the ranks in S.
Perform communication with neighbour MPI ranks based on Halo as described in Algorithm 3.

2.4 Performance Investigation

As described in Section 2.2, the width of the halo region is a parameter that may be tuned to achieve lower communication times. Wider halos enable a wider range of particle velocities and hence particle displacements to be handled using a local communication approach and also reduce time spent in an expensive global communication patter. To investigate the potential performance improvement we create a communication bound benchmark that transports particles across a 2-D square domain with periodic boundary conditions. The particle velocities are sampled from a normal distribution in each dimension such that approximately 10% of particles have a velocity greater than $v_{\text{cutoff}} = E/(4\delta t)$ where E is the domain edge length and δt is the time step size. At each time step the position of each particle is simply updated using the carried velocity and no interactions with either a field or other particles occur.

We apply a common domain decomposition approach where the MPI ranks are arranged in a Cartesian grid known as a Cartesian communicator. This Cartesian grid is used as the imposed global Cartesian mesh required to map a position in space to an owning MPI rank. Furthermore we also use this Cartesian grid as a replacement for the unstructured mesh which would exist when the particle system is coupled with a FEM library. Each time the number of MPI ranks is doubled the number of inter-rank boundaries over which particles must be communicated also doubles.

Algorithm 3 Overview of neighbour based bookkeeping operation using halo information.

Require: List of remote MPI ranks S this rank could send particles to, ie. the owning ranks of cells in the halo region on this rank.
Require: List of remote MPI ranks R that this rank could receive from, ie. remote ranks that possess a halo cell owned by this rank.
for remote rank $r \in R$ **do**
  Start non-blocking receive from r for receive particle count.
end for
for remote rank $s \in S$ **do**
  Start non-blocking send to s to send particle count.
end for
Pack particle data to send to remote ranks.
Wait for all non-blocking receives to complete and allocate memory to receive.
for remote rank $r \in R$ **do**
  Start non-blocking receive from r for particle data.
end for
for remote rank $s \in S$ **do**
  Start non-blocking send to s to send particle data.
end for
Unpack received particle data and remove sent particle data.

Hence for a fixed global particle count the communication cost of the simulation increases linearly with the number of MPI ranks. We assume that performance of the high performance interconnect used in HPC facilities at best increases linearly with the number of MPI ranks.

In practice the message latency of such a network is unlikely to improve as the number of MPI ranks and messages increases but, depending on the network topology, the total bandwidth may improve as the number of ranks increases. For small message sizes, such as the data for an individual particles, message latency is typically the dominant metric for communication performance. Thus we expect that in the best case scenario the time taken to perform one time step of the simulation is a constant.

We compare three different approaches that transfer particle data between MPI ranks. In the most simple scenario we disable all halo cells and perform communication of particle data using only the global method described in Algorithm 2. This global-only case represents a communication pattern that does not attempt to exploit the problem structure to reduce global communication. In second method we fix the halo region to have a width equal to one MPI rank which enables neighbour based communication to the immediate neighbours (including in the diagonal direction) of each rank. As the number of MPI ranks increases the physical width of this halo will reduce. Hence in the strong scaling limit we expect the performance of this method to converge to the global-only approach. In the final method we allow the halo region to grow in size as the number of MPI ranks increases such that the width of the halo is at least $v_{\text{cutoff}}\delta t$ in all directions. Hence approximately 90% of particle data should be communicated via the neighbour based communication for any number of MPI ranks.

In Figure 2 we present the time taken per simulation step against the number of MPI ranks. These

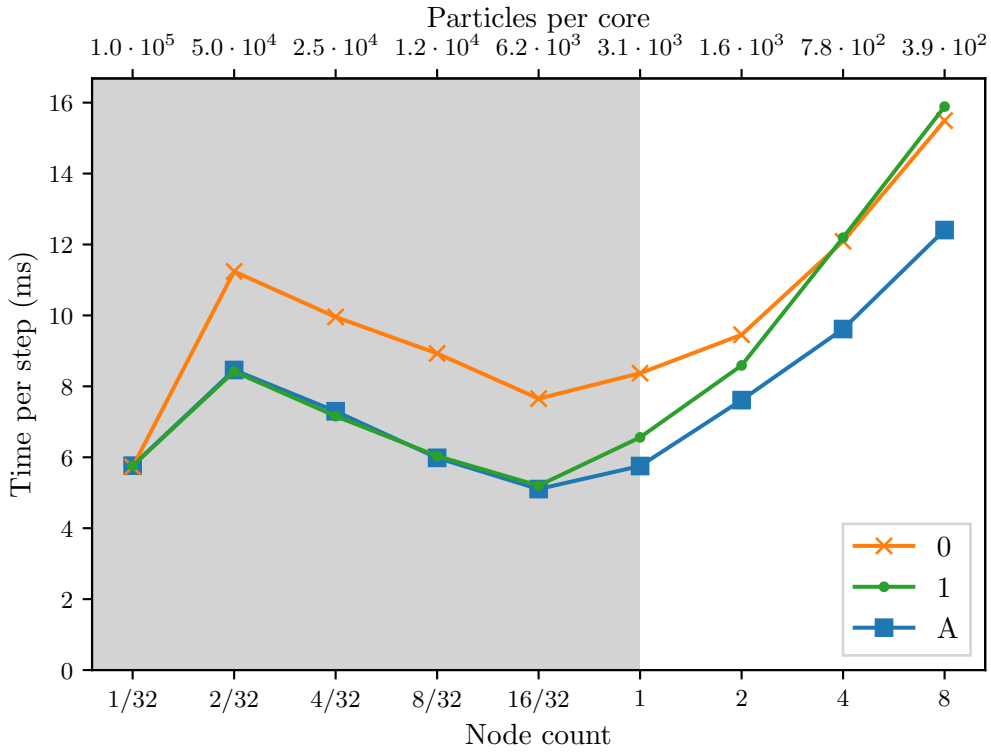


Figure 2: Time per simulation step against node count with one MPI rank per cpu core. "0" indicates the case without neighbour based communication. "1" indicates a halo of width 1 MPI rank. "A" indicates an adaptive halo of width at least $v_{\text{cutoff}}\delta t$. Grey region indicates intra-node scaling.

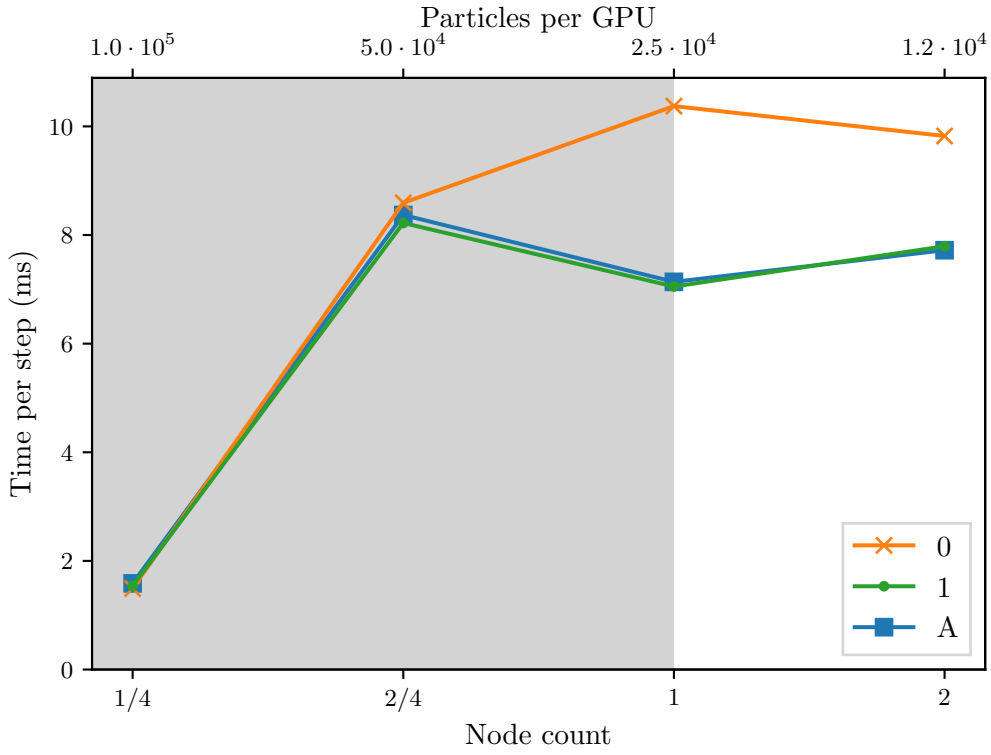


Figure 3: Time per simulation step against node count with one MPI rank per A100 GPU. "0" indicates the case without neighbour based communication. "1" indicates a halo of width 1 MPI rank. "A" indicates an adaptive halo of width at least $v_{\text{cutoff}}\delta t$. Grey region indicates intra-node scaling.

CPU scaling results were produced using the CSD3-Peta4 HPC facility where nodes consist of two Intel Xeon Gold 6142 (16-core Skylake) CPUs and a Intel Omni-Path interconnect. We use Julia 1.6.2 for all Julia code along with Intel MPI 2020.4. We observe that the global-only approach is slower in all cases than the neighbour based approach with an adaptive halo width. We also observe that as the number of MPI ranks increases the time taken per step for a fixed halo of width one rank converges to the global-only method as expected. Between 1 and 8 nodes we observe that the time per simulation step approximately doubles and we notice an expected increase in time at the transition from one to two nodes.

Finally in Figure 3 we repeat the same communication experiment presented in Figure 2 with the modification that the simulation is performed on GPU hardware. These results are produced from the CSD3-Wilkes3 facility where nodes consist of two AMD EPYC 7763 (Zen 3 64-core) CPUS and four Nvidia A100 GPUs. The CPU software environment was also used for these GPU results. By using the DSL described in M4.2[3] the only modification required to execute the simulation on GPU hardware, as opposed to CPU hardware, is a change of target device - essentially one line of code. Particle data and particle based operations are then performed on the GPU device as opposed to the CPU. The purpose of these timing results is to demonstrate the portability of the DSL first introduced in M4.2 and the timing results should be considered preliminary as GPU

optimisations are yet to be implemented.

3 $1d1v$: Two Stream Instability

It is necessary to gain understanding of the confluence of PIC, FEM and semi-implicit methods. To this end a toy PIC code was written to self-consistently solve for the electrostatic field using FFTs whilst evolving the trajectories of Dirac delta function shaped particles. The charge deposition of particle quantities to finite element basis functions and the ”opposite” operation, reading FEM fields to particle positions, was the main learning point. The key is to ensure that the particle quantity was apportioned proportional to the basis functions' heights at the particle position and ensuring that either the basis functions partition unity (or this is corrected for if they do not partition unity).

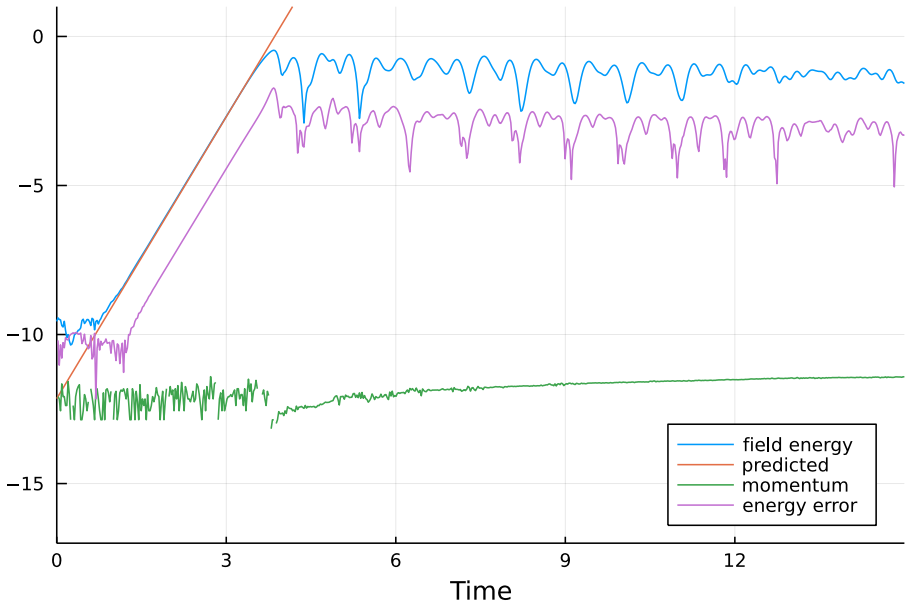


Figure 4: Energy, momentum and associated errors plotted against time for the two stream instability test case.

In Figure 4 we plot the two stream instability results for the 1-D physical space 1-D velocity space implementation. This figure shows that the two stream growth rate is accurately reproduced, momentum and charge are conserved to machine precision, and energy is conserved adequately.

4 Summary

A representative simulation of the scrape off layer should incorporate a wide range of different physical phenomena that require particular attention to be modelled both efficiently and adequately. In this report we investigated two implementation details that are relevant to both the plasma species and the neutral species. We demonstrated in Section 2 that combining a neighbour based communication pattern with a long-range global communication method offers a significant reduction in the time taken to communicate particles between MPI ranks. Optimisation of inter-rank communication is critical to achieving good parallel scaling on modern HPC hardware and hence it is important to consider these implementation details at the design stage of project NEPTUNE. In Section 3 we demonstrated how we were able to look at interactions between particles and FEM.

Acknowledgement

The support of the UK Meteorological Office and Strategic Priorities Fund is acknowledged.

References

[1] Dmitriy V. Borodin, Friedrich Schluck, Sven Wiesen, D M Harting, Petra Boerner, Sebastijan Brezinsek, Wouter Dekeyser, Stefano Carli, Maarten Blommaert, Wim Van Uytven, Martine Baelmans, Bert Mortier, Giovanni Samaey, Yannick Marandet, Paul Genesio, Hugo Bufferand, Egbert Westerhof, Jorge Gonzalez, Mathias Groth, Andreas Holm, Niels Horsten, and Huw Leggate. Fluid, kinetic and hybrid approaches for neutral and trace ion edge transport modelling in fusion devices. *Nuclear Fusion*, 2021.

[2] J.A. Wesson. *Tokamaks, 3rd Edition*. Clarendon Press, Oxford, 2003.

[3] W. Saunders, J. Cook, and W. Arter. 2-D Model of Neutral Gas and Impurities. Technical Report CD/EXCALIBUR-FMS/0061-M4.2, UKAEA, 2021. `https://github.com/ExCALIBUR-NEPTUNE/Documents/blob/main/reports/ukaea_reports/CD-EXCALIBUR-FMS0061-M4.2.pdf`.