IMPACT OF LOAD BALANCING ON PARALLEL PERFORMANCE WITH HAAR WAVELETS ANGULAR ADAPTIVITY

Ioannis Nikiteas[1], **Steven Dargaville**[1], **Christopher C. Pain**[1],
Paul N. Smith[2], **Richard P. Smedley-Stevenson**[1,3]
[1]Applied Modelling and Computation Group, Imperial College London
London SW7 2AZ, UK

[2]ANSWERS Software Service, Wood PLC
Kimmeridge House, Dorset Green Technology Park,
Winfrith Newburgh, Dorchester, Dorset, DT2 8ZB
[3]AWE, Aldermaston, Reading, RG7 4PR, UK

ioannis.nikiteas17@imperial.ac.uk

ABSTRACT

This paper tests the performance gain of performing load balancing with angular adaptivity, that has been discretised using Haar wavelets. Load balancing on highly refined angular meshes resulted into a reduction in the runtime of approximately 50% on a local machine. The time spent load balancing was found to be proportional to the number of unknowns of the problem, with the rate of the proportionality being dependant on the number of processing cores used. Strong scaling performance of 97% was achieved in 240 cores, dropping to 50% for 480 cores, for a mesh of 405,000 triangular elements. The ratio of nodes to halos indicates that as we adaptively refine, the load balancing algorithm produces an increasing number of partitions with fewer nodes and larger halos. Hence, destroying parallel scaling by performing communications instead of a solving the linear system.

KEYWORDS: dynamic load balancing, angular adaptivity, haar wavelets

1. INTRODUCTION

There exist two different approaches for modelling radiation transport phenomena, through the use of either deterministic or stochastic numerical methods. Common deterministic numerical methods such as Finite Element use a mesh and nodes to represent the geometry of the problem. Given that the accuracy of the solution is correlated to the fineness of the mesh; in cases where increased accuracy is required in the solution of the problem, the mesh can be refined locally which is referred to as adaptive refinement.

Parallel radiation transport codes decompose the original mesh into smaller partitions and distribute its elements across multiple processing cores. A performance bottleneck is created when the parallel performance of the code has to be maintained but simultaneously the mesh has to be

adaptively refined multiple times. By refining a partition, the number of degrees of freedom (NDOFs) in it increases and hence, the computational work that has to be performed on the adapted partition increases as well. To reduce the computational work imbalance across the partitions, the partitions have to be redrawn through a process referred to as load balancing. A single load balance requires extensive communication between all of the decomposed partitions in order to obtain the load imbalance across the entire computational domain. Then certain elements of the mesh have to be redistributed across the newly generated partitions, which can be a time consuming process. Therefore, by increasing the order of adaptive refinement the time spent load balancing on a mesh is also inevitably increased.

This paper demonstrates the runtime improvements that have been achieved by performing load balancing on an angular adaptive problem, where the angular domain has been discretised with Haar wavelets. The code used for the radiation transport simulations is FETCH2, developed by the AMCG in Imperial College London.

2. THEORETICAL BACKGROUND

2.1. Angular Discretisation: Haar Wavelet

Haar wavelets are a piecewise constant family of functions, which are hierarchical and compactly supported [1, 2]. Using the two dimensional non-standard Haar wavelet discretisation found in [3], it is possible to create an angular domain discretisation equivalent to a hierarchical P_0 – DGFEM, which forces a constant azimuthal angle in polar coordinates, shown in Figure 1a. With the term adaptive order or adaptive step being used to signify the number of mesh refinements that have occurred on the angular mesh.

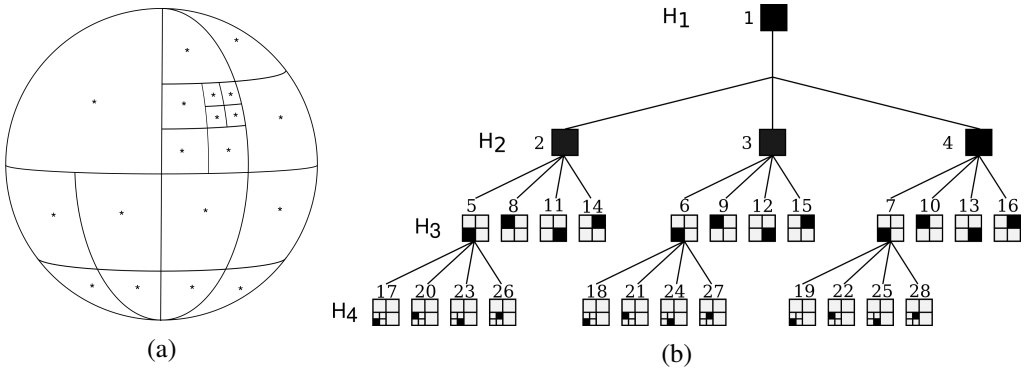


Figure 1: (a) Adapted fourth order (H_4) Haar wavelet discretisation. (b) Quadtree representation of Haar wavelets in one octant of the unit sphere. Shaded regions correspond to the area where the wavelets have support over.

As a result of the hierarchical nature of Haar wavelets and the forced equivalency between P_0 – DGFEM and the Haar wavelets, the discretisation can be represented with a hierarchical data structure such as quad-trees. Figure 1b depicts a quad-tree, corresponding to an octant (quadrant in 2D) of the discretised unit sphere from Figure 1a. In addition, the quad-trees carry unique identifiers referred to as angle numbers representing all present elements in the nodes.

Notably, this specific discretisation not only does it allow for anisotropic angular resolution, but simple arithmetic operations on the angle numbers can indicate the relative position of an element in the angular discretisation or the parent element that the refined angle number is derived from; simplifying the process of coarsening the angular mesh while maintaining $\mathcal{O}(n)$ scalability in both runtime and memory consumption as shown by Dargaville et al. [3], with n corresponding to the number of angles present in the node.

2.2. Spatial Discretisation: Sub-Grid Scale

For the spatial discretisation a sub-grid scale (SGS) formulation has been used [2, 4–6], where the spatial mesh is divided into a "fine" and a "coarse" finite element mesh.The "fine" mesh holds the Discontinuous Galerkin (DG) representation of the problem, while the "coarse" mesh holds the Continuous Galerkin (CG).

The use of SGS results into a stabilised spatial discretisation, capable of handling heavy advection problems. Moreover, SGS has been shown to reduce the number of Degrees of Freedom (DOF), when compared to conventional DG methods, resulting into reduced memory requirements [2].

2.3. Adaptivity Algorithm

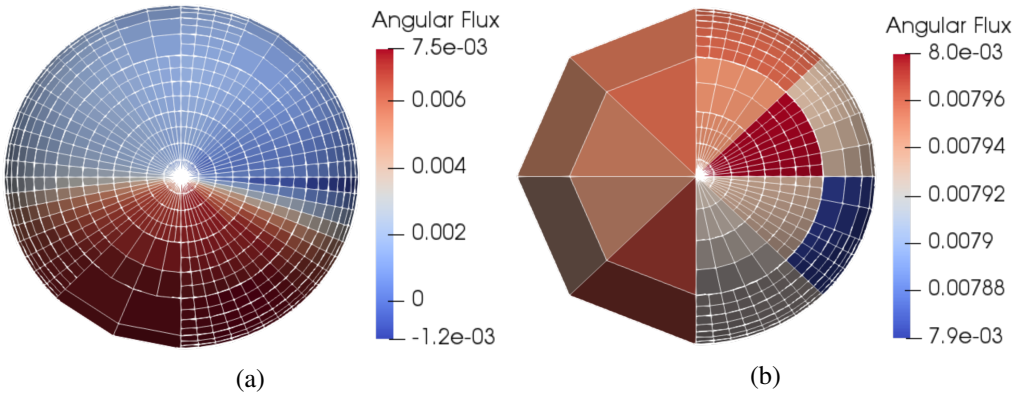
In this paper we only investigate the usage of regular Haar wavelet adaptivity in parallel, load balanced simulations. A brief overview of the regular error metric is given below, but for a more detailed description along with a Haar wavelet goal-based error metric see [3].

The Haar wavelet discretisation simplifies the adapting process by making use of the compact support and hierarchy, allowing for local refinement and coarsening of the mesh, without need to interpolate between adapt orders. Moreover, our wavelet space exploits norm-equivalence and cancellation properties [3, 7]. The norm-equivalence suggests a direct relationship between the norm of the Haar wavelet coefficients and the norm of the discretised function. Therefore, small Haar wavelet coefficients have small contributions to the norm of the approximated function. The cancellation properties refer to the Haar coefficients being small when the function is smooth, for a given discretisation order, over the wavelet support [3, 7].

Thus the norm-equivalence and cancellation properties enable the use of a threshold value to drive the adaptive refinement process, with large coefficients triggering refinement and small coefficients causing coarsening [3, 7]. Adapting the angular discretisation results in different NDOFs at each spatial node, which is depicted in Figure 2, where two different spatial nodes from the same spatial mesh are shown after having adapted differently in angle.

The problem solved is described by an exact solution ψ_{exact}, which has a residual R and an error ϵ, with $R(\psi_{\text{exact}}) = 0$ and $\epsilon \equiv \psi_{\text{exact}} - \psi$ respectively. As previously mentioned, Haar wavelets enable the use of thresholding. Hence, given a threshold t the approximation $\mathbf{e}$ of the exact error can then be calculated by $\mathbf{e} = \epsilon \approx |\psi|/t$ [3].

Important to note is the iterative method used by FETCH2, which is a FGMRES preconditioned by a matrix-free multigrid solver with asynchronous capabilities for performing matrix-vector multiplications [3, 8, 9]. To perform the linear solves on the partition boundaries, certain nodes from



(a) (b)

Figure 2: Top-down view of two spatial nodes and their corresponding azipolar 2D adapted Haar wavelet discretisations.

neighbouring partitions are required. These shared nodes are often referred to as halos. The halo nodes are also where the asynchronous communications occur during a solve [3].

2.4. Dynamic Load Balancing Configuration

Dynamic Load Balancing (DLB) is used when performing adaptive refinement to minimise the computational imbalances introduced by adaptivity and consequently reduce the total runtime time. To redistribute the computational load across all processors a partitioning tool is required. FETCH2 has been configured with the graph partitioner ParMETIS [10].

The selection of ParMETIS was based on two requirements; short partitioning time and high mesh quality. The DLB algorithm needs to be relatively cheap to compute compared to the main computation, since periodic repartitioning may be required. Furthermore, the partitioning algorithm should be able to yield high quality partitions, since higher quality partitions may result into shorter runtimes. Graph partitioners have been documented to return the highest quality partitions out of all other DLB algorithms, however they tend to be computationally expensive [11]. As it is showcased in Section 3 the time spent load balancing is only a minuscule fraction of the total runtime, enabling us to use ParMETIS without a significant impact on the total runtime.

The adapted angular discretisation is load balanced by assigning a series of weights to all nodes of the spatial mesh. The weights are representative of the adapted angular discretisation at each spatial node and can therefore be used as a metric of the work across the computational domain.

In more detail, the DLB algorithm after every adaptive refinement calculates the load imbalance τ^*, by counting the size of angular expansions (NDOFs) present in a processor, for all processors. The load imbalance can then be defined as the ratio between the processor with the most unknowns (most heavily loaded) divided by the processor with the least (least heavily loaded). Load balancing occurs if the load imbalance exceeds the imbalance tolerance τ. The default tolerance for all calculations is set to $\tau = 1.05$, equivalent to tolerating a 5% load imbalance in the computations.

Furthermore, before repartitioning the mesh, weights need to be calculated for all spatial nodes by

taking the ratio of the angular expansion at every node and dividing with the greatest angular expansion in the entire spatial mesh. ParMETIS internally approximates the communications cost by counting the number of edge-cuts in the graph of the partitions, therefore, no additional weighting is required on the nodes to account for the inter-processor communications cost of load balancing. Supplying the resulting list of weights to ParMETIS returns the load balanced spatial meshes. In the improbable scenario that the partitioning algorithm has generated poor results, the load imbalance is calculated again to check whether subsequent domain decompositions are required in a single adaptive refinement step. Thus the order of execution of FETCH2 for a single adapt can be represented by the flowchart shown in Figure 3.

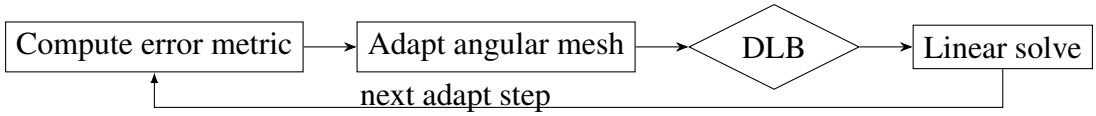


Figure 3: FETCH2 flowchart for angular adaptivity and load balancing.

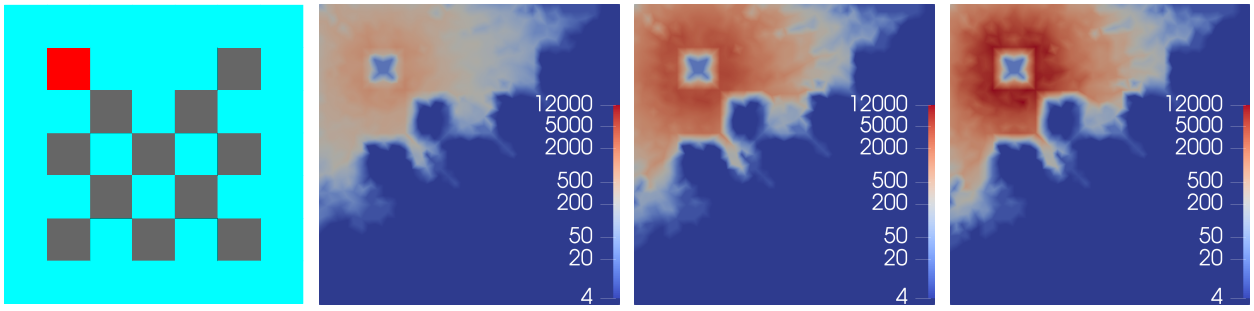
3. RESULTS

3.1. Brunner Lattice problem

Testing the performance and scalability of the load balancing algorithm was performed by creating a highly asymmetrical case of the problem first proposed by Brunner [12], displayed in Figure 4a. The test case includes a neutron source located in the top left corner, whilst surrounded by strong absorbers and scatters.

The model was allowed to adapt the angular mesh to an adapt order of ten. The effects of increased orders of adaptivity on the angular domain are displayed in Figures 4b–4d. Adapting the angular mesh, refines the angular discretisation in areas where the neutron flux varies rapidly, such as in the case of nodes near a neutron source. The finely discretised angular domain of a node contains an increased number of unknown variables, hence, the computational work required to obtain a solution on such node is increased. Consequently, mesh partitions containing multiple nodes that have been refined require a greater amount of computational work and are therefore imbalanced. The load imbalances caused by adaptivity when load balancing are disabled are provided in the caption of Figures 4b–4d. All the data have been obtained using a value of $\mathbf{e}$ equal to 0.001, where $\mathbf{e}$ is scaled against the maximum scalar flux.

Initially, the DLB was tested on a local 4 core machine with a mesh containing 3,200 triangular elements and 6,600 Continuous Degrees of Freedom (CDOFs), where it was observed that the total runtime of FETCH2, was reduced by a factor of two when DLB was enabled, as seen in Figure 6. Moreover, the enlarged section of Figure 6 displays how there exist certain imbalance tolerances that result into slightly better runtimes, but also how the performance of load balancing algorithm is generally insensitive to the imbalance tolerance τ. A visualisation of load balancing is provided by Figures 5a and 5b, where Figure 5a corresponds to the initial domain decomposition of the mesh and Figure 5b corresponds to the domain decomposition at adapt order ten, where all four partitions require the same computational work to evaluate the solution at their nodes. In both cases, the halo nodes are not displayed.



(a) (b) (c) (d)

Figure 4: (a) Schematic of Brunner lattice. ● is a pure scatterer, with a macroscopic total cross section of $\Sigma_t = 1\text{cm}^{-1}$, ● is a pure absorber, with $\Sigma_t = 10\text{cm}^{-1}$, ● is the neutron source with strength $1\text{cm}^{-3}\,\text{s}^{-1}$ and $\Sigma_t = 10\text{cm}^{-1}$. (b)–(d) Visual representation of angular adaptivity at different adapt orders with the colourmap displaying in a logarithmic scale the number of angles placed at a point in space. When load balancing is disabled the load imbalance τ^* increases exponentially after every adapt step; (b) adapt = 6, $\tau^* = 94$ (c) adapt = 8, $\tau^* = 409$ (d) adapt = 10, $\tau^* = 698$.

(a) (b)

Figure 5: Spatial domain decomposition that results from load balancing the angular domain at (a) adapt order 1, initial mesh decomposition (b) adapt order 10.

The above problem was then run on ARCHER, the UK's Tier 1 supercomputer (CRAY XC30), in order to conduct a strong scaling study. A strong scaling study is performed by testing how a problem of fixed size varies as the number of processors is altered. Two spatial meshes were tested both using triangular elements, with mesh1, having 210,000 elements and 420,000 CDOFs initially, and mesh2 having 405,000 triangular elements and 810,000 of CDOFs. The exponential increase of the CDOFs can be seen in Figure 7, where every data point on a line corresponds to the CDOFs at an adapt step.

Additionally, Figure 7 shows two important relations; first, the time spent load balancing is directly proportional to the NDOFs in the discretisation and second, the rate of this proportionality is dependant on the number of cores used.

The load balancing algorithm performs a spatial mesh decomposition, however, only the angular

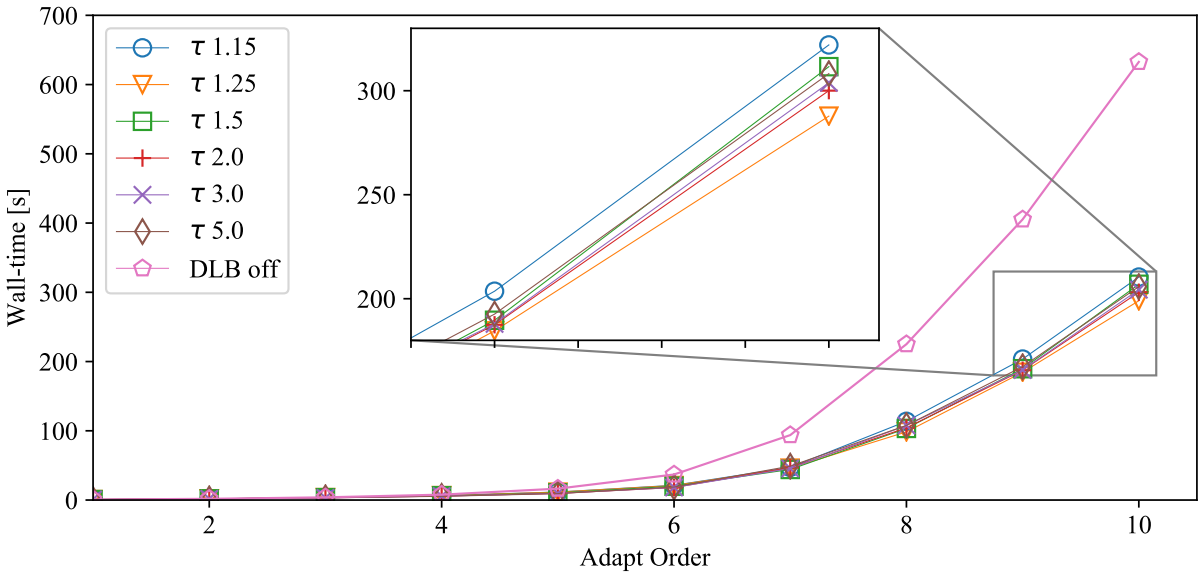


Figure 6: Total runtime as a function of the angular adaptive refinement run on a local 4 core machine.

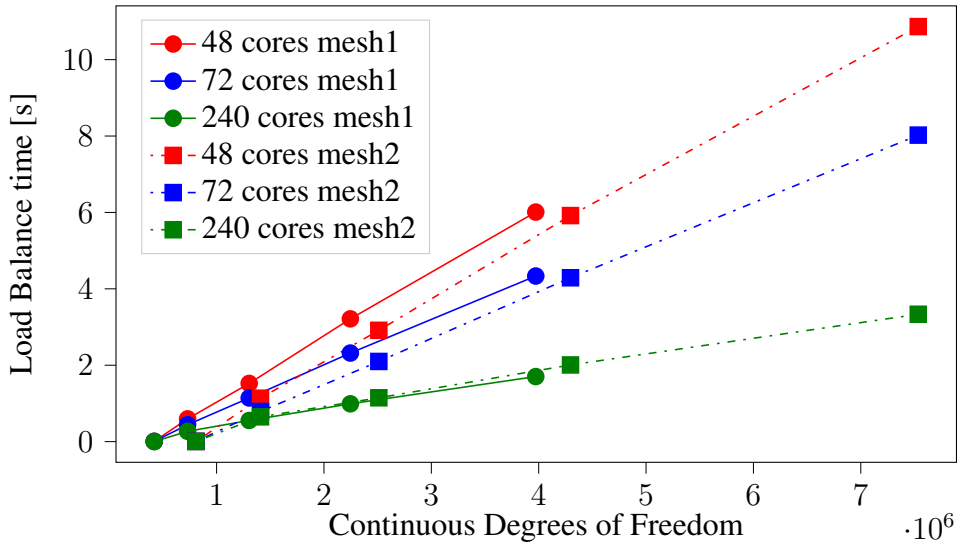
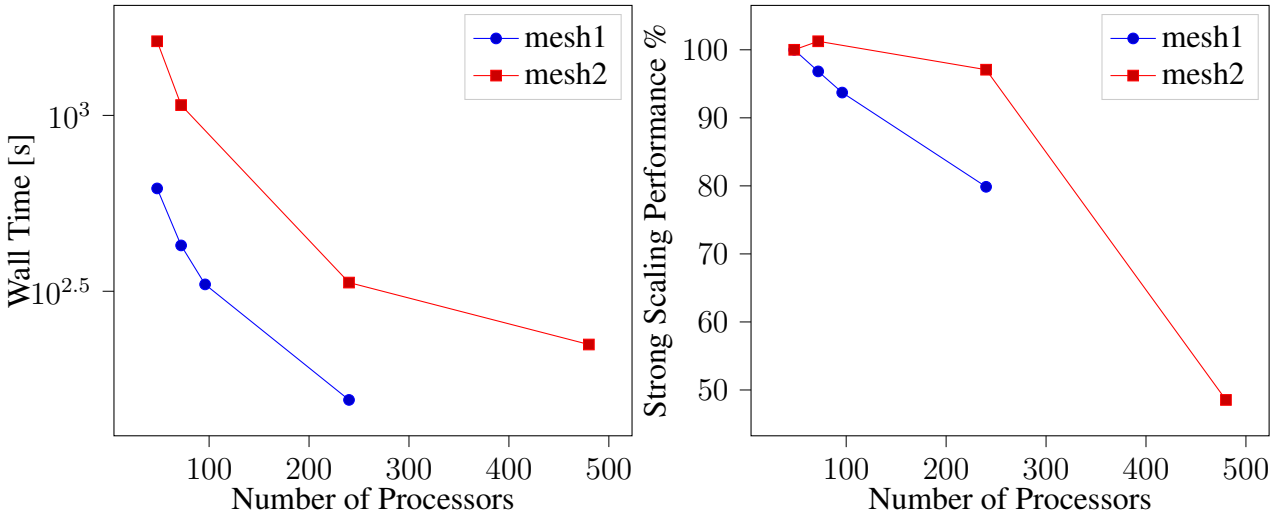


Figure 7: Time spent load balancing as function of the CDOFs, shows proportionality with the number of processors and the number of CDOFs.

mesh is adapted. We expect that if the number of processors and/or the adapt order continue to increase a point will be reached where the spatial decomposition, in some cores, will contain only a small number of nodes and a large number of halo nodes. This is expected to be a generic problem, true for all problems that angular adaptivity with DLB is applied, since the angular adaptivity drives the DLB to produce asymmetrical spatial mesh decompositions.



(a) (b)

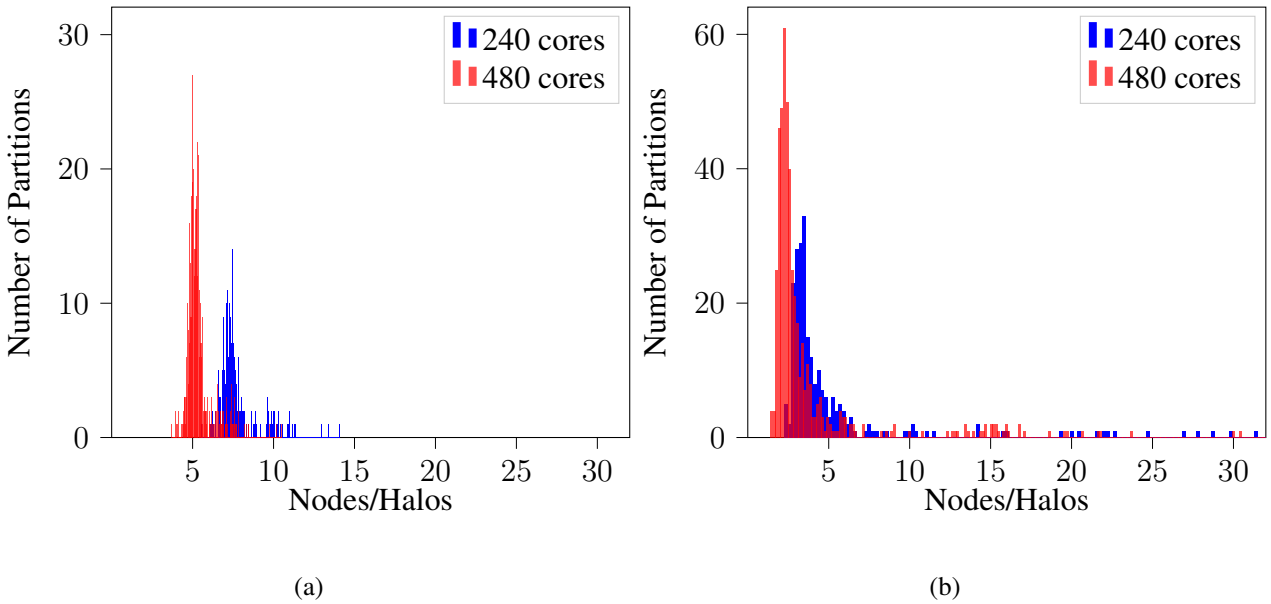
Figure 8: Strong scaling performance with the load balancing enabled. (a) Wall time performance. (b) Strong scaling performance.

Even though after a load balance all the partitions require equal computational work, partitions with few owned nodes and large halos are expected to be spending a considerable amount of time performing parallel communications, which in turn would result in a loss of parallel performance. The loss of parallel performance can be seen in Figure 8. With Figure 8a illustrating the exponential decay in improvement of runtime performance as the number of cores increases, especially in the case of 240 to 480 cores. Similarly, Figure 8b, showcases the strong scaling performance for the Brunner problem. With mesh1, experiencing a steady decline in strong scaling performance, reaching 80% for 240 cores and mesh2 sustaining a 97% strong scaling performance for 240 cores, but then rapidly decreasing to 50% for the 480 core case.

Additionally, a useful parallel performance metric is the ratio between the number of owned nodes in a partition and its halo size. A large ratio in a partition corresponds to good parallel performance, while a smaller ratio implies more halo nodes and hence, a greater amount of time spent doing parallel communications. Plotting the distributions of the ratios in the 240 and 480 core cases for mesh2 provides further insight as to whether our predictions for the node/halo size impacting parallel performance are true. First, Figure 9a depicts the initial spatial mesh decomposition. Both the 240 and 480 core runs are similar. A major mode exists corresponding to the majority of the partitions which have approximately equal ratios of nodes/halos. A second mode is present in the distribution with a higher node/halo ratio and fewer occurrences corresponding to the boundary partitions. The mesh boundaries have reduced halos and hence result to greater node/halo ratios. Finally, certain partitions end up with fewer or greater number of neighbouring partitions than what the average number is. The different number of nearest neighbours results into different halo patterns and consequently marginally different node/halo ratios, causing the distribution to disperse. After five adapt steps the major mode of the nodes/halos distribution has shifted noticeably to a smaller ratio and the minor mode has dispersed to higher ratios causing the distribution to become

more bimodal as seen from Figure 9b. This trend amplifies as we adapt and/or increase the number of cores, resulting in the 480 core node/halo distribution into having ~ 150 partitions with a node/halo ratio of 2, on the fifth adapt step, as seen in Figure 9b.

It is important to emphasize, that all the partitions in Figure 9 are balanced with respect to their computational work, but only the partitions of Figure 9a are (mostly) evenly sized between each other.



(a) (b)

Figure 9: Distribution of the ratio between owned nodes in a partition over its halo nodes for mesh2. (a) Distribution at adapt step 0. (b) Distribution at adapt step 5.

4. CONCLUSIONS

Load balancing an adaptively refined angular Haar wavelet discretisation shows promising results for the runtime performance of the neutron transport code FETCH2, by reducing by half the wall-time in local machines. In distributed memory machines, good (97%) strong scaling was achieved by using meshes that had average node/halo ratios in the last adapt step greater or equal to 4. This result is of course partially specific to the type the iterative method (FGMRES) and mesh partitioning tool (ParMETIS) used, but nevertheless minimising the size of the halos will yield improved runtime performance in most cases. Finally, due to the strong scaling performance greatly decreasing in the limited strong scaling tests that were run, future work will involve improving the performance of FETCH2 by investigating alternate communication patterns in our iterative solver for partitions with small node/halo ratios.

REFERENCES

[1] A. Adam, A. G. Buchan, M. D. Piggott, C. C. Pain, J. Hill, and M. A. Goffin. "Adaptive Haar wavelets for the angular discretisation of spectral wave models." *Journal of Computational Physics*, **volume 305**, pp. 521–538 (2016).

[2] A. G. Buchan, A. S. Candy, S. R. Merton, C. C. Pain, J. I. Hadi, M. D. Eaton, A. J. H. Goddard, R. P. Smedley-Stevenson, and G. J. Pearce. "The Inner-Element Subgrid Scale Finite Element Method for the Boltzmann Transport Equation." *Nuclear Science and Engineering*, **volume 164**(2), pp. 105–121 (2010).

[3] S. Dargaville, A. G. Buchan, R. P. Smedley-Stevenson, P. N. Smith, and C. C. Pain. "Scalable angular adaptivity for Boltzmann transport." *Journal of Computational Physics* (2019).

[4] T. J. Hughes, G. R. Feijóo, L. Mazzei, and J.-B. Quincy. "The variational multiscale method—a paradigm for computational mechanics." *Computer Methods in Applied Mechanics and Engineering*, **volume 166**(1-2), pp. 3–24 (1998).

[5] T. J. Hughes, G. Scovazzi, P. B. Bochev, and A. Buffa. "A multiscale discontinuous Galerkin method with the computational structure of a continuous Galerkin method." *Computer Methods in Applied Mechanics and Engineering*, **volume 195**(19-22), pp. 2761–2787 (2006).

[6] A. S. Candy. *Subgrid scale modelling of transport processes*. Ph.D. thesis, Imperial College London (2008).

[7] A. Cohen, W. Dahmen, and R. DeVore. "Adaptive Wavelet Techniques in Numerical Simulation." In *Encyclopedia of Computational Mechanics*. John Wiley & Sons, Ltd, Chichester, UK (2004).

[8] Y. Saad. "A Flexible Inner-Outer Preconditioned GMRES Algorithm." *SIAM Journal on Scientific Computing*, **volume 14**(2), pp. 461–469 (2005).

[9] S. Dargaville, A. G. Buchan, R. P. Smedley-Stevenson, P. N. Smith, and C. C. Pain. "Scalable angular adaptivity for Boltzmann transport." *Journal of Computational Physics* (2019). URL http://arxiv.org/abs/1901.04929.

[10] G. Karypis and V. Kumar. "Multilevel Graph Partitioning Schemes." *[ICPP'95] International Conference on Parallel Processing*, **volume 3**, pp. 113–122 (1995).

[11] B. Hendrickson and K. Devine. "Dynamic load balancing in computational mechanics." *Computer Methods in Applied Mechanics and Engineering*, **volume 184**(2-4), pp. 485–500 (2000).

[12] T. A. Brunner. "Forms of Approximate Radiation Transport." Technical Report SAND2002-1778, Sandia National Laboratories, Albuquerque, New Mexico (2002).