Massively Parallel Simulations of Binary Black Hole Intermediate-Mass-Ratio Inspirals

Milinda Fernando(student)*, Hari Sundar (advisor)†, David Neilsen ‡, Hyun Lim § and Eric Hirschmann ¶
*milinda@cs.utah.edu, †hari@cs.utah.edu, ‡david.neilsen@byu.edu, §hyun.lim@byu.edu, ¶ehirsch@physics.byu.edu,

I. Introduction

We present a portable and highly-scalable framework (DENDRO-GR) that targets problems in the astrophysics and numerical relativity communities. The recent detection of Gravitational Waves (GW) has revolutionized multi-messenger astronomy and produced several exciting discoveries. Current GW observations have been limited to observations of binary mergers of roughly equal mass. This has to a large part been due to the inability of existing computational relativity codes to simulate Intermediate Mass Ratio Inspirals (IMRIs) with a mass-ratio (q) greater than 10. Our approach, DENDRO-GR framework is able to simulate IMRIs with mass ratio as high as $q = 100$. We have designed novel scalable algorithms to enable efficient simulations for such experiments and demonstrate excellent weak scalability up to $131K$ cores on ORNL's Titan for binary mergers for mass ratios up to 100. The key contributions of this work include:

Wavelet Adaptive GR. To the best of our knowledge, the other comparable codes use simple models of adaptivity, i.e.,structured adaptivity, block adaptivity, or logically uniform grids [1], [2], [3], [4]. We present a novel computational GR framework (DENDRO-GR) which uses octree-based Adaptive Multiresolution(AMR) grids, where the adaptivity is determined by wavelet expansion [5], [6], [7], [8], [9] of the functions represented in the underlying grid. We refer this as Wavelet Adaptive Multiresolution (WAMR). This is the first highly adaptive computational fully relativistic—i.e., including the full Einstein equations—code with an arbitrary localized adaptive mesh. For example for a mass ratio of $q = 1$, we use approximately `7x` fewer degrees of freedom for the same accuracy compared to the block adaptivity (via Carpet [10]).

Automatic symbolic code-generation. Given the complexity of the Einstein equations, we have developed an automatic code generation framework for GR using `SymPy` that automatically generates architecture-optimized codes. This greatly improves code portability, use by domain scientists and the ability to add additional constraints and checks to validate the code.

TREESEARCH. We developed a new parallel search algorithm TREESEARCH, to improve the efficiency of octree meshing. We also developed efficient `unzip` and `zip` operations to allow the application of the core computational kernels on small process-local regular blocks. This allows greater performance as well as greater performance portability.

II. Methodology

The Figure 3 on the poster presents an overview of our approach.

A. TREESEARCH: *Efficient search operations on octrees*

By *mesh generation* we refer to the process of building data structures required to perform numerical computations on a topological 2:1 balanced complete octree. We build two additional data structures referred as octant to octant (O2O) & octant to nodal(O2N) maps. O2O contains neighborhood information at an octant level (i.e. neighbor octants of a given octant), while O2N contains for a given octant its corresponding nodes. In order to build those maps, we need to perform search operations on the adaptive octree. The state-of-the-art approaches [11] uses SFC based ordering operator combined with binary searches. The key drawback of this approach is random, unstructured memory accesses leading to bad memory performances especially the octree size is significantly large. Assuming we need to search k keys on size n octree, the time complexity for binary search approach would be $\mathcal{O}(k\log(n))$. In our approach TREESEARCH we start with the root node, perform bucketing for each key to the corresponding child if root and recurse until we hit a leaf octant(see Figure 1). The time complexity of TREESEARCH can be written as $\mathcal{O}(k\log(n))$, which is similar to the binary search approach, but TREESEARCH approach perform $k\log(n)$ streaming passes compared to random accesses leading to better memory performances.

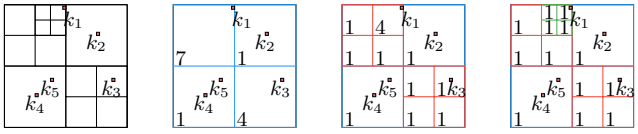


Fig. 1: For a given ordered octree τ and a set of keys (leftmost figure), TREESEARCH performs the traversal in a top-down order over the set of keys, while flagging k_2, k_4, k_5 at the level 1 split, k_3 at level 2 split, and k_1 at level 3 split.

B. *unzip & zip: Finite difference computations on adaptive grids*

The main objective of the *unzip* and *zip* operations is to enable stencil computations on adaptive octrees. Any adaptive octree can be considered as a union of regular sub-octrees which we referred as blocks or *octree to block decomposition*. To perform stencil computations on these blocks, it is required to have neighborhood region surrounding the block which

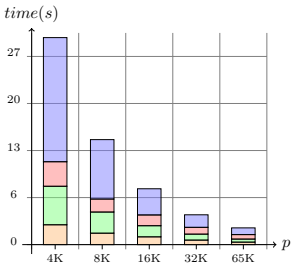
referred as *padding*. The block with surrounding padding region is referred as *unzip* representation(see Figure 6 on the poster). The stencil and other update operations are only performed on the block as the padding is read-only. At the end of the update, the simulation variables are *zipped* back, i.e., injected back to the *zipped* representation. Note that several key operations such as RK update & inter-process communications operate using the *zip* representation, and are efficient (see Figure 3 on the poster).

C. Symbolic interface and code generation

The Einstein equations are a set of non-linear, coupled, partial differential equations. On discretization, one can end up with 24 or more equations with thousands of terms. Writing, optimizing and maintaining code for this is very challenging. Sustainability and keeping it relevant for new architectural changes are additional difficulties. To address these issues, we have developed a symbolic interface to DENDRO-GR. We leverage symbolic python (`SymPy`) as the backend for this along with the python package `cog` to embed python code within our application-level `C++` code(see symbolic code generation on the poster).

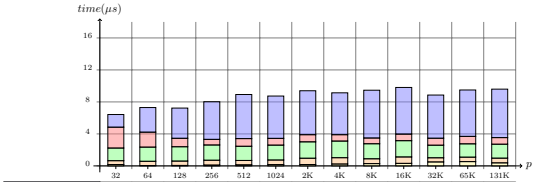
III. RESULTS

In this section, we present results on scalability studies performed on DENDRO-GR (see Figures 2 & 3) and comparison study with the EINSTEIN TOOLKIT(see Figure 4) which also provides an insight on the efficiency of WAMR grid compared to structured/block adaptivity. We also present a comparison study between TREESEARCH & binary search approach (see Figure 5 on the poster).



	4K	8K	16K	32K	65K
communication	18.31	8.76	3.86	1.85	0.99
unzip	3.62	1.92	1.59	1.00	0.66
rhs	5.68	3.12	1.64	0.88	0.45
derivatives	2.93	1.69	1.15	0.67	0.37
Total time (s)	30.5	15.5	8.2	4.4	2.5

Fig. 2: Strong scaling results in ORNL's Titan for a single RK step (averaged over 10 steps) with derivative computation (`deriv`), right hand side(`rhs`) computation, `unzip` cost and communication cost (`comm`) for a fixed problem size of $10.5B$ unknowns where the number of cores ranging from $4,096$ to $65,536$ cores on 4096 nodes. Note that for strong scaling results re-meshing is disabled in order to keep the problem size fixed.



	32	64	128	256	512	1024	2K	4K	8K	16K	32K	65K	131K
communication	1.65	3.21	3.94	4.91	5.75	5.52	5.73	5.46	6.23	6.08	5.63	6.05	6.32
unzip	2.72	1.95	1.10	0.73	1.00	0.88	0.93	0.83	0.75	0.86	0.91	0.97	0.86
rhs	1.65	1.85	1.87	1.99	1.85	1.94	2.14	2.17	1.96	2.12	1.64	1.74	1.82
derivatives	0.50	0.55	0.57	0.63	0.57	0.60	0.82	0.82	0.63	0.84	0.54	0.58	0.61
wavelets	0.18	0.05	0.06	0.11	0.14	0.17	0.18	0.25	0.30	0.33	0.52	0.56	0.40
Total time	6.7	7.6	7.5	8.4	9.3	9.1	9.8	9.5	9.9	10.2	9.2	9.9	10.0
Total dofs (zipped)	48.6M	92.9M	187.7M	342.1M	694.0M	1.4B	2.8B	5.6B	11.0B	21.7B	46.4B	93.9B	205.8B

Fig. 3: Weak scaling results in ORNL's Titan for $RK/(dof/p)$ (averaged over 10 steps) where RK, dof, p denotes the time for single RK step, degrees of freedom, and number of cores respectively, where the approximate grain size of 1.536M unknowns where the number of cores ranging from 32 to $131,072$ cores on 8192 nodes where the largest problem having 206 Billion unknowns.

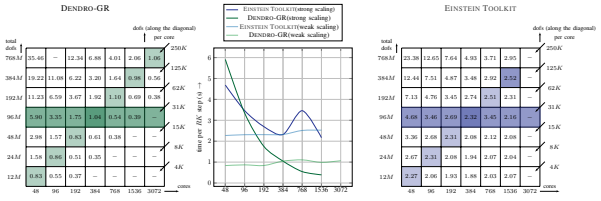


Fig. 4: Comparison between EINSTEIN TOOLKIT and DENDRO-GR without factoring in adaptivity (i.e. both EINSTEIN TOOLKIT & DENDRO-GR support uniform grids.). For a fixed tolerance, we expand the domain for a $1:1$ mass-ratio simulation such that both EINSTEIN TOOLKIT and DENDRO-GR have roughly the same number of dofs. We present both weak and strong scaling results using both codes. On the left table are results from DENDRO-GR and from EINSTEIN TOOLKIT on the right. In the middle, we plot a representative strong and weak scaling curve for each code. The DENDRO-GR scaling is plotted in green (lighter shade for weak) and blue for EINSTEIN TOOLKIT. The corresponding data entries are also marked in the tables. Note that the rows represent strong scaling and the diagonal entries represent weak scaling results and runtime is reported in seconds(s).

REFERENCES

[1] Einstein Toolkit, http://einsteintoolkit.org.

[2] D. Neilsen, S. L. Liebling, M. Anderson, L. Lehner, E. O'Connor *et al.*, "Magnetized Neutron Stars With Realistic Equations of State and Neutrino Cooling," *Phys.Rev.*, vol. D89, no. 10, p. 104029, 2014.

[3] B. Bruegmann, J. A. Gonzalez, M. Hannam, S. Husa, U. Sperhake, and W. Tichy, "Calibration of Moving Puncture Simulations," *Phys. Rev.*, vol. D77, p. 024027, 2008.

[4] T. Yamamoto, M. Shibata, and K. Taniguchi, "Simulating coalescing compact binaries by a new code SACRA," *Phys. Rev.*, vol. D78, p. 064054, 2008.

[5] O. V. Vasilyev, S. Paolucci, and M. Sen, "A multilevel wavelet collocation method for solving partial differential equations in a finite domain," *J. Comput. Phys.*, vol. 120, pp. 33 – 47, 1995.

[6] O. V. Vasilyev and S. Paolucci, "A dynamically adaptive multilevel wavelet collocation method for solving partial differential equations in a finite domain," *J. Comput. Phys.*, vol. 125, pp. 498–512, 1996.

[7] S. Paolucci, Z. J. Zikoski, and D. Wirasaet, "WAMR: An adaptive wavelet method for the simulation of compressible reacting flow. Part I. Accuracy and efficiency of algorithm," *J. Comput. Phys.*, vol. 272, pp. 814 – 841, 2014.

[8] S. Paolucci, Z. J. Zikoski, and T. Grenga, "WAMR: An adaptive wavelet method for the simulation of compressible reacting flow. Part II. The parallel algorithm," *J. Comput. Phys.*, vol. 272, pp. 842 – 864, 2014.

[9] M. Holmström, "Solving hyperbolic pdes using interpolating wavelets," *SIAM J. Sci. Comput.*, vol. 21, no. 2, pp. 405–420, 1999.

[10] Carpet, an AMR driver for Cactus, http://www.carpetcode.org.

[11] J. Rudi, A. C. I. Malossi, T. Isaac, G. Stadler, M. Gurnis, P. W. J. Staar, Y. Ineichen, C. Bekas, A. Curioni, and O. Ghattas, "An extreme-scale implicit solver for complex pdes: Highly heterogeneous flow in earth's mantle," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, ser. SC '15. New York, NY, USA: ACM, 2015, pp. 5:1–5:12. [Online]. Available: http://doi.acm.org/10.1145/2807591.2807675