A Heterogeneous, In Transit Approach for Large Scale Cellular Modeling

AYMAN YOUSEF, Duke University, USA
AMANDA RANDLES (ADVISOR), Duke University, USA

CCS Concepts: • **Human-centered computing** → **Scientific visualization**; • **Applied computing** → *Computational biology*; • **Computing methodologies** → *Simulation evaluation*; Parallel algorithms.

Additional Key Words and Phrases: in situ, high performance computing, visualizaton

1 INTRODUCTION

The field of in silico cellular modeling has made notable strides in number of cells that can be simultaneously modeled [4]. The availability of modern, high-performance systems has allowed researchers to conduct simulations at much larger scales, leading to more precise representations of critical physiological phenomena (immune response, molecular metabolism [3] [6]). While computational capabilities have grown exponentially, I/O performance has lagged behind. To address this issue, in situ-based methodologies, in which data is analyzed while in memory (Figure 1), have gained traction. These methods are not without drawbacks, stemming from overhead introduced with coupling the analysis/simulation [1]. To overcome these limitations, in transit methods have emerged, where data is handed off to separate visualization "workers" to decouple the tasks of visualization and simulation [5]. In this work, we present an in-transit method that enables concurrent simulations on separate MPI threads, harnessing the heterogeneity of leadership-class systems to overcome I/O limitations and enhance overall efficiency.

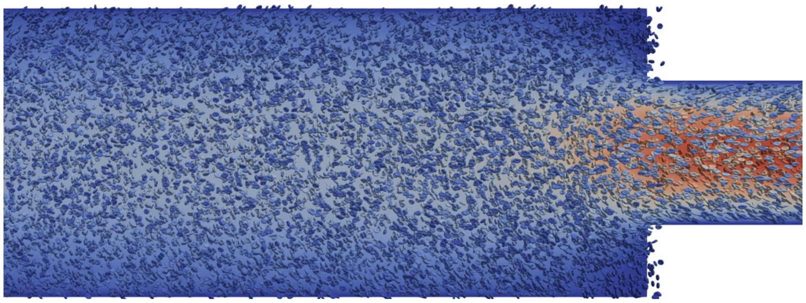

Fig. 1. Example in situ visualization of a cell simulation. By maintaining accurate simulation data on a separate thread, we facilitate data analysis in transit whilst data remains in memory.

Authors' addresses: Ayman Yousef, ayman.yousef@duke.edu, Duke University, Durham, North Carolina, USA, 43017-6221; Amanda Randles (ADVISOR), amanda.randles@duke.edu, Duke University, Durham, North Carolina, USA, 43017-6221.

Manuscript submitted to ACM

2 METHODOLOGY

In this study, we propose an in-transit algorithm designed to decouple data generation and visualization tasks during simulations. To evaluate the effectiveness of our approach, we implement and assess the framework within the context of the massively parallel fluid solver HARVEY [8, 9].

2.1 HARVEY: Massively parallel fluid solver

HARVEY utilizes stenciling, a technique characterized by spatial decomposition of a domain. "Halo regions" or "halos" are exchanged between neighboring tasks during the communicative portion to supply the data necessary to complete upcoming arithmetic operations. Within HARVEY, red blood cells (RBCs) are modeled as fluid-filled membranes embedded in a Lagrangian mesh. HARVEY uses the immersed boundary method (IBM) [7], a common approach to couple cell/fluid solvers. Algorithmic updates within these halo regions in both terms of fluid and cell necessitate perpetual communication with all neighboring tasks to correctly update points. The complex representation of RBCs within HARVEY makes this a memory-intensive undertaking.

2.2 Loosely Coupled In Transit

The traditional in transit workflow is characterized by passing data to a separate resource to perform analysis, allowing undisturbed simulation progression. Our framework differs from the standard approach, using HARVEY's stencil structure to avoid a full data transfer. Stenciling allows us to employ a version in which halo data is transferred instead, enabling access to the full simulation state without needing to perform any iterative communication between processes and at a cheaper data movement cost. We can be split our framework two-fold: the recording and replay components.

2.2.1 Recording. Within HARVEY, n-dimensional buffers are initialized in anticipation of communication for IBM-related operations. Separate buffers exist for each array of data being passed and are dynamically resized to fit the volume of data to be passed and received. The buffers are cleared after each iteration to be used again. Our in-transit framework allocates separate n +-dimensional "halo" buffers to accommodate the storage of multiple time steps worth of information. During each time step, "halo" data is loaded into these buffers and remains in memory until accessed. We co-opt the halo data exchanged between processes during HARVEY's fluid- and cell-specific portions.

2.2.2 Replay. Each MPI process maintains two simulation states in parallel. This scheme is achieved through OpenMP threading on separate portions of the node. The primary simulation is run on the CPU and GPU as usual, recording data throughout. A separate secondary simulation is managed on only the CPU, loading data from the predefined buffers at the appropriate time steps to circumvent the need for any communications. Therefore, MPI processes within this simulation are independent outside of any enforced collective operations such as visualization. Analysis routines are executed in situ from the secondary thread using its version of the simulation data, allowing the main simulation to proceed unperturbed. The supply of halo data ensures that the secondary simulation maintains synchronicity with the primary simulation.

3 RESULTS AND DISCUSSION

We focus our attention on the recording portion of our methodology, as it is the most costly operation among the two. In Figure 2, we present our findings on the weak scaling of the recording portion within our in-transit framework. Throughout the experiment, we performed up to 8,192 tasks, resulting in an overhead ranging from 2% to 6%. Interestingly, we did not observe a clear correlation between task count and overhead. By leveraging pre-existing data transfers, we managed the overhead effectively, especially when packing buffers with existing data. Although the overhead is not entirely negligible, it remains comparable to the results of previous work done with the fluid-only version of a similar framework [2].

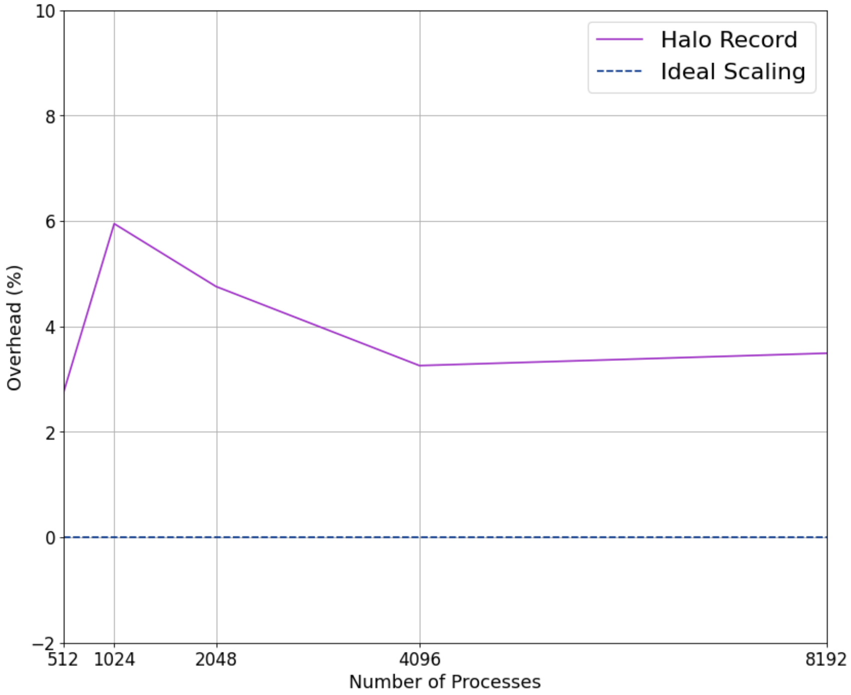

Fig. 2. Weak scaling of in-transit recording iteration time across input size. The average number of cells per processor is kept consistent across task counts. Cell data is recorded at each time step into preallocated halo buffers

4 SUMMARY

In the context of exascale computing, in situ-based data exploration offers significant advantages due to its low storage requirements during data creation. This study introduces a framework that preserves these advantages while eliminating the need for interval simulation stops or complete data transfers across nodes. We provide a detailed description of this framework and analyze it to evaluate the associated overhead.

REFERENCES

[1] James Ahrens, Sébastien Jourdain, Patrick O'Leary, John Patchett, David H Rogers, and Mark Petersen. 2014. An image-based approach to extreme scale in situ visualization and analysis. In *SC'14: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*. IEEE, 424–434.

[2] Jeff Ames, Silvio Rizzi, Joseph Insley, Saumil Patel, Benjamín Hernández, Erik W Draeger, and Amanda Randles. 2019. Low-overhead in situ visualization using halo replay. In *2019 IEEE 9th Symposium on Large Data Analysis and Visualization (LDAV)*. IEEE, 16–26.

[3] Mariagrazia Belfiore, Marzio Pennisi, Giuseppina Aricò, Simone Ronsisvalle, Francesco Pappalardo, et al. 2014. In silico modeling of the immune system: cellular and molecular scale approaches. *BioMed Research International* 2014 (2014).

[4] Leopold Grinberg, Joseph A Insley, Vitali Morozov, Michael E Papka, George Em Karniadakis, Dmitry Fedosov, and Kalyan Kumaran. 2011. A new computational paradigm in multiscale simulations: Application to brain blood flow. In *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis*. 1–5.

[5] James Kress. 2020. In-line vs. in-transit in situ: Which technique to use at scale? (2020).

[6] Joshua A Lerman, Daniel R Hyduke, Haythem Latif, Vasiliy A Portnoy, Nathan E Lewis, Jeffrey D Orth, Alexandra C Schrimpe-Rutledge, Richard D Smith, Joshua N Adkins, Karsten Zengler, et al. 2012. In silico method for modelling metabolism and gene product expression at genome scale. *Nature communications* 3, 1 (2012), 929.

[7] Charles S Peskin. 2002. The immersed boundary method. *Acta numerica* 11 (2002), 479–517.

[8] Amanda Randles and Efthimios Kaxiras. 2014. A Spatio-temporal Coupling Method to Reduce the Time-to-Solution of Cardiovascular Simulations. In *2014 IEEE 28th International Parallel and Distributed Processing Symposium*. 593–602. https://doi.org/10.1109/IPDPS.2014.68

[9] Amanda Peters Randles, Vivek Kale, Jeff Hammond, William Gropp, and Efthimios Kaxiras. 2013. Performance analysis of the lattice Boltzmann model beyond Navier-Stokes. In *2013 IEEE 27th International Symposium on Parallel and Distributed Processing*. IEEE, 1063–1074.