Integrated Parallelization of Computations and Visualization for Large-scale Applications

Preeti Malakar[1]*, Vijay Natarajan[2]*†, Sathish S. Vadhiyar[2]†
*Department of Computer Science and Automation
†Supercomputer Education and Research Centre
Indian Institute of Science, Bangalore, India
preeti@csa.iisc.ernet.in, vijayn@csa.iisc.ernet.in, vss@serc.iisc.ernet.in

***Abstract*—Critical applications like cyclone tracking and earthquake modeling require simultaneous high-performance simulations and online visualization for timely analysis. Faster simulations and simultaneous visualization enable scientists provide real-time guidance to decision makers. In this work, we have developed an integrated user-driven and automated steering framework that simultaneously performs numerical simulations and efficient online remote visualization of critical weather applications in resource-constrained environments. It considers application dynamics like the criticality of the application and resource dynamics like the storage space, network bandwidth and available number of processors to adapt various application and resource parameters like simulation resolution, simulation rate and the frequency of visualization. We formulate the problem of finding an optimal set of simulation parameters as a linear programming problem. This leads to 30% higher simulation rate and 25-50% lesser storage consumption than a naive greedy approach. The framework also provides the user control over various application parameters like region of interest and simulation resolution. We have also devised an adaptive algorithm to reduce the lag between the simulation and visualization times. Using experiments with different network bandwidths, we find that our adaptive algorithm is able to reduce lag as well as visualize the most representative frames.**

I. Introduction

High-performance and high-fidelity numerical simulation is paramount to many scientific and engineering applications like weather modeling and computational fluid dynamics. These simulations involve large-scale computations and generate large amount of data. Visualization enriches the process of scientific discovery by helping the scientist to easily comprehend the data. Simultaneous visualization of the simulation data can reduce the end-to-end simulation-visualization time. In our work, we perform online and remote visualization, where the simulation and visualization are simultaneously performed at different locations. Such a simulation-visualization model can enable geographically distributed scientists to collaboratively analyze the visualization and provide expert opinion on the occurrence of critical events.

A. Simultaneous simulation and visualization

High simulation rates on modern-day processors combined with high I/O bandwidth [1]–[3] can lead to rapid accumulation of data at the simulation site. We have shown in our work that it is important to consider these resource constraints for continuous simulation and smooth visualization [4]. We use linear programming to optimize the simulation speed and simulation output frequency, given the various resource constraints like storage space, computation speed, network bandwidth and I/O bandwidth. This ensures smooth progress of simulation and visualization.

B. Reconciling algorithmic and user-driven steering

Online visualization can allow the user to give feedback to the ongoing simulation. Such a computational steering framework for high-performance simulations of critical applications needs to take into account both the steering inputs of the scientists and the criticality needs of the application. The framework should analyze the combined effect of the user-specified parameters and the resource constraints on the smooth progress of the simulation. We have shown in [5] how our adaptive steering framework can both algorithmically steer the simulations and can be used by the scientist to modify the simulation parameters. Our framework InSt combines user-driven steering with automatic tuning of application parameters based on resource constraints and the criticality needs of the application like minimum progress rate of simulation to determine the final parameters for the simulations.

C. Efficient online visualization

Another challenge related to remote online visualization of critical weather applications is to minimize the *lag* between the time when the simulation produces an output frame and the time when the frame is visualized. The lag depends on the output frequency determined by the framework and the network bandwidth. It is important to reduce the lag between the simulation and visualization times so that the scientists are able to get *on-the-fly* view of the simulation. Also, for critical applications like tsunami prediction, the faster the simulation output reaches the visualization site, the better are the chances of taking active measures on time. We address the critical issue of reducing this lag by adapting to the available resource parameters and the simulation output. We have developed algorithms to minimize the lag between simulation and visualization and concurrently visualize important events in the simulation. Using experiments with different network configurations, we find that our adaptive algorithm strikes

[1]Number of years in PhD: 3.5
[2]Advisor

a good balance in reducing the lag and visualizing most representative frames.

II. Related Work

The conventional approach of visualizing simulation output is *offline* post-processing. However, for critical applications like weather forecasting, *online* visualization strategies play an important role. In-situ visualization has been extensively studied in the recent times [6]–[9]. In [6], [7] the authors use a common data structure for both simulation and visualization. A drawback of this approach is that the same data structures are often not efficient for both simulation and visualization. Moreover, this tight-coupling leads to stalling of simulation when visualization is performed. Since high-fidelity simulations demand more computations and more resources than visualization, it is undesirable to stall the simulation. Moreover, the longer a scientist interacts with the data, the longer is the simulation stalled.

Another approach of in-situ visualization is to use shared physical memory by the simulation and visualization phases. This alleviates the time taken to write data onto the secondary memory but requires huge amount of physical memory for accurate simulations. In [8], the authors use 1 TB of shared memory in their experimental setup. A long-running high-resolution simulation with high output frequency executed on modern-era petaflop supercomputers is highly likely to suffer from unavailability of memory to store the simulation output. Also, the longer the scientist interacts with the current time-step data, the lesser will be the time taken to overflow the physical memory. An important drawback of all the above approaches is that the visualization expert cannot interact with the full mesh, going back and forth in time. In our work, we aim to perform online visualization and continuous simulation, without the requirement of a huge physical memory and giving full flexibility to the scientist to interact with the full mesh, going back and forth in time.

For an efficient online visualization across networks with differing bandwidths, our framework reactively selects representative frames to be sent from the simulation to the visualization site. Wang et al. [10] derive an importance measure for each spatial block in the joint feature-temporal space of the data based on the formulation of conditional entropy. Their histogram calculation takes 19 hours for a 960 x 660 x 360 volume for 222 time steps with block size of 48 x 33 x 18. Such a large amount of time for selection of important frames from a running simulation is clearly unsuitable for efficient online visualization. Patro et al. [11] have presented a method to measure saliency in molecular dynamics simulation data. Their keyframe selection method for molecular dynamics simulations considers the atom positions to determine saliency of the atoms in a time step and then aggregate information to find saliency of the time step. Their saliency function is highly specific to molecular dynamic simulations where the changes in atomic positions between consecutive time steps of a simulation are dominated by Brownian motion and interesting and purposeful molecular conformational changes occur only over larger time scales. In our work, we propose a generic frame selection algorithm.

III. Our approach

Simultaneous and continuous visualization for user-guided simulation of critical applications require a robust middleware for efficient disk space management and processor allocation to prevent disk overflow despite limited resources and huge amount of simulation output. To minimize the end-to-end delay of the entire simulation-visualization pipeline, we propose a framework that considers both application and resource dynamics to adapt various simulation and visualization parameters like simulation resolution and frequency of output for visualization.

In [4], we show that it is important to consider resource constraints like storage space, computation speed and network bandwidth between simulation and visualization sites for smooth simulation and continuous visualization. We developed an adaptive framework that considers these resource constraints and application dynamics to automatically tune the simulation and visualization parameters like the number of processors and the frequency of visualization.

In [5], we proposed an enhanced adaptive integrated steering framework, InSt, that reconciles between algorithmic and user-driven steering for continuous simulation and visualization. InSt determines the runtime parameters like the number of processors for simulation and the output frequency of simulation output, based on resource characteristics and user input, within the constraints related to continuous visualization, acceptable output frequency, I/O bandwidth, disk space and network speed.

A schematic of InSt is shown in Figure 1. It consists of *a simulation process* to perform coordinated simulations, *a visualization process* for online remote visualizations and user-driven steering, *frame sender and receiver daemons* to transfer frames from simulation to visualization sites and *an application manager* that determines the application configuration for simulations. The application manager periodically invokes a decision algorithm, which determines the number of processors, and the frequency of simulation output for execution of simulations for a given resolution of simulation, the network bandwidth between the simulation and visualization sites, the available disk space at the simulation site, and the minimum progress rate of simulations desired by the user.

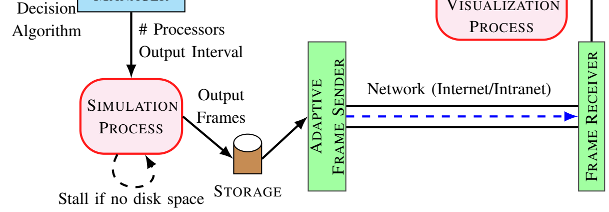


Fig. 1. InSt: Adaptive Integrated Steering Framework

A. Algorithmic Steering

The objective of the decision algorithm is to maximize the rate of simulations and to enable continuous visualization with maximum temporal resolution, but these objectives are contradictory. Maximum temporal resolution can be achieved by increasing the frequency of output of data. However, increasing the frequency of output can decrease the rate of simulations due to increase in number of writes to the disk and can lead to rapid consumption of storage, eventually stalling the simulations. On the other hand, decreasing the output frequency can increase the simulation rate, but will result in visualization of fewer frames. Unlike traditional scheduling algorithms that maximize simulation rates, our decision algorithm may have to sometimes *slow down* the simulations, since faster simulations can lead to faster consumption of storage if the network to the visualization site is slow. Thus we formulated our problem as an optimization problem that primarily attempts to maximize the simulation rate subject to the resource constraints of computation rate, I/O bandwidth, the disk space, the network speed and the minimum progress rate of simulations. Our decision algorithm is a constrained linear programming problem to obtain the number of processors and the frequency of output for simulations. Since we want the best possible throughput of the simulation in spite of the resource constraints, we express the objective of our optimization problem as $minimize\ t$ where t is the execution time to solve a time step and the constraints are related to limited disk space available, continuous simulation and visualization and minimum progress rate of simulation.

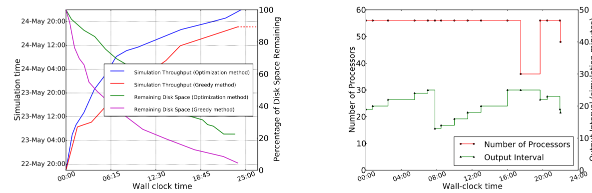
B. User-driven Steering

INST performs user-driven steering to enable the user to explore the simulation and instantaneously impact the simulation. Our framework allows a user to create a finer resolution simulation, modify simulation resolution, output interval and the minimum progress rate of simulation. INST checks the feasibility of running the simulations with the user inputs, advises the users of alternate options if not feasible, and invokes the decision algorithm with the user inputs and resource parameters if feasible.

C. Efficient Online Visualization

It is imperative for simulation output to be visualized as soon as the data is produced, to reduce the end-to-end simulation-visualization turnaround time, especially for critical applications. However, if the network bandwidth is low and the simulation speed is high, then the visualization times can lag far behind the simulation times. This is because the frames may take longer time to be transferred and hence the number of frames simulated in that time will be higher. Thus there will be increasing number of queued frames at the simulation site. It is easy to observe that with increasing queue of pending frames, the lag for successive frames keeps increasing. We propose an adaptive solution to this problem.

INST tries to adapt to the lag between the simulation and visualization time. If the lag is unacceptable by the end user, our *adaptive* algorithm sends a set of representative frames from the queued frames at the simulation site. This ensures that the framework adapts to varied network bandwidths to reduce the lag. The higher the network bandwidth, the lesser will be the number of queued frames and more will be the total number of representative frames selected during the simulation. These representative frames are chosen by a linear time modified k-means algorithm for temporal clustering [12]. However, since the transfer time of a frame is proportional to the amount of data in the frame, sending the whole frame may still lead to unacceptable lag. In such cases, we send reduced information per frame, so that the salient data is retained in the frames and the frames are sent within acceptable lag limits. This satisfies the goal of high temporal resolution so that the important events are not missed.



(a) Simulation progress (Left y-axis) and disk usage (Right y-axis) for optimization method and greedy heuristic

(b) Adaptivity of INST showing variation in number of processors (Left y-axis) and output interval (Right y-axis)

Fig. 2. Inter-country simulation across very low bandwidth

IV. EXPERIMENTS AND RESULTS

We elaborate few of the obtained results using weather simulation for a critical weather application, namely, tracking of cyclone *Aila*. We used a popular weather forecast model, WRF (Weather Research and Forecasting Model) [1] for the simulation and VisIt [13] for visualization of WRF output. With decreasing pressure at the *eye* of the cyclone, InSt refines the weather simulation for more accurate output. INST also dynamically spawns a finer-resolution simulation within the ongoing simulation. We have developed a plug-in for VisIt to *directly read* WRF output files, eliminating the cost of post-processing before data analysis. Visualization was carried out on a workstation in Indian Institute of Science (IISc) with a Intel(R) Pentium(R) 4 CPU 3.40 GHz and an NVIDIA graphics card GeForce 7800 GTX.

A. Simulation throughput and Adaptivity

Figure 2 shows the results for an experimental setup with low bandwidth of 60 Kbps between the simulation and visualization sites. The simulations were carried out in *moria* cluster at University of Tennessee, Knoxville. Figure 2(a) compares the simulation rate for our optimization-based decision algorithm with a simple greedy heuristic which runs on the maximum number of processors and outputs every time step initially. The greedy heuristic reactively modifies the simulation parameters with decreasing disk space but still fails to complete the simulation due to disk overflow, as

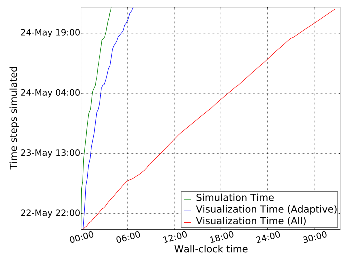


Fig. 3. Simulation and visualization times for medium-bandwidth setup

shown by the red dotted lines in the figure. The optimization-based method has a higher throughput than the greedy method because of optimal selection of parameters. It can be observed that the greedy approach consumes the disk faster and finally the disk is full as shown by the magenta curve. However, the optimization approach is able to complete the simulation without disk overflow. Figure 2(b) illustrates the adaptation of the number of processors and the output interval of the simulations by INST in order to continuously simulate despite low network bandwidth, limited disk space of 100 GB and dynamic refinement of simulation resolution from 24 km to 10 km.

B. Lag reduction

Figure 3 illustrates the effect of the *adaptive* lag reduction algorithm on the visualization times. This simulation was conducted on the *ggblr* cluster at the Centre for Development of Advanced Computing, Bangalore and the available network bandwidth to the visualization site at IISc was 16 Mbps. The green curve shows the simulation times and the red curve shows the visualization times for the default approach of sending all the queued frames. It can be noted that the difference between the intersection points of the green and the red curves with a horizontal line will give the instantaneous lag. The blue curve shows the visualization times for the representative frames sent. The clustering algorithm ensures that the representative frames are able to capture the temporal information. This approach also reduces the lag for an efficient online visualization. It can be seen from the figure that at 23 May 13:00 hours on the y-axis, the lag between the visualization time and the simulation time has been reduced by 90% by the adaptive algorithm.

V. REMAINING OBJECTIVES

In the future, we plan to investigate research challenges related to multiple simultaneous simulations and visualizations. Multiple finer-level simulations can be useful to simultaneously visualize several important events at a greater detail. We would like to extend the framework to support concurrent visualization requirements from different geographically distributed users. In this direction, we are currently exploring the scheduling algorithms for multiple simulations. We intend to improve the scalability of weather simulation models on really large number of processors. We also plan to apply our techniques for critical applications from diverse domains.

VI. CONCLUSIONS

As we enter the exascale era, it is important to efficiently perform high-performance scientific simulations and visualization, inspite of the different growth rates in the computation speeds and the memory and network bandwidths. In this work, we presented an adaptive integrated steering framework for simulation and visualization of critical applications like cyclone tracking across various resource configurations. Our framework adapts to the resource characteristics like the available disk space, network bandwidth, computation speed and dynamically alters the application parameters like simulation resolution and output interval for continuous visualization. Our framework enables a scientist to steer an ongoing simulation and modify the simulation parameters on-the-fly. Since the user may be unaware of the resource constraints, we propose a novel combination of automatic tuning and user-driven steering where the framework tries to satisfy the user inputs within the resource constraints. We also proposed an adaptive algorithm to perform efficient online visualization in order to minimize the simulation-visualization lag.

REFERENCES

[1] J. Michalakes and et al., “WRF Nature Run,” in *Proceedings of the 2007 ACM/IEEE conference on Supercomputing*, 2007.

[2] H. Yu and et al., “High Performance File I/O for The Blue Gene/L Supercomputer,” in *Proceedings of the 12th International Symposium on High-Performance Computer Architecture*, 2006.

[3] S. Lang, P. H. Carns, R. Latham, R. B. Ross, K. Harms, and W. E. Allcock, “I/O performance challenges at leadership scale,” in *Proceedings of the 2009 ACM/IEEE conference on Supercomputing*, 2009.

[4] P. Malakar, V. Natarajan, and S. Vadhiyar, “An Adaptive Framework for Simulation and Online Remote Visualization of Critical Climate Applications in Resource-constrained Environments,” in *Proceedings of the 2010 ACM/IEEE conference on Supercomputing*, 2010.

[5] ——, “InSt: An Integrated Steering Framework for Critical Weather Applications,” in *ICCS 2011: Proceedings of the International Conference on Computational Science*, 2011.

[6] K.-L. Ma, C. Wang, H. Yu, and A. Tikhonova, “In Situ Processing and Visualization for Ultrascale Simulations,” *Journal of Physics (Proceedings of SciDAC 2007 Conference)*, vol. 78, 2007.

[7] T. Tu and et al., “From Mesh Generation to Scientific Visualization: an End-to-End Approach to Parallel Supercomputing,” in *Proceedings of the 2006 ACM/IEEE conference on Supercomputing*, 2006, p. 91.

[8] D. Ellsworth and et al., “Concurrent Visualization in a Production Supercomputing Environment,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 5, pp. 997–1004, 2006.

[9] N. Fabian and et al., “The ParaView Coprocessing Library: A scalable, general purpose in situ visualization library,” in *IEEE Symposium on Large Data Analysis and Visualization*, 2011.

[10] C. Wang, H. Yu, and K.-L. Ma, “Importance-Driven Time-Varying Data Visualization,” in *IEEE Transactions on Visualization and Computer Graphics*, 2008.

[11] R. Patro, C. Y. Ip, and A. Varshney, “Saliency Guided Summarization of Molecular Dynamics Simulations,” in *Scientific Visualization: Advanced Concepts*, 2010, pp. 321–335.

[12] J. B. MacQueen, “Some Methods for Classification and Analysis of MultiVariate Observations,” in *Proceedings of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1, 1967, pp. 281–297.

[13] H. R. Childs, E. Brugger, K. S. Bonnell, J. S. Meredith, M. Miller, B. Whitlock, and N. Max, “A Contract Based System For Large Data Visualization,” in *IEEE Visualization*, 2005.