Jul 12th, 10:50 AM - 11:10 AM

In situ Data Visualization of ACME Land Model Simulations

Yu Pei
Oak Ridge National Laboratory, peiy@ornl.gov

Benjamin Hernandez
Oak Ridge National Laboratory, hernandezarb@ornl.gov

Dali Wang
Oak Ridge National Laboratory, wangd@ornl.gov

Brian Smith
Oak Ridge National Laboratory, smithbe@ornl.gov

Scott Atchley
Oak Ridge National Laboratory, atchleyes@ornl.gov

See next page for additional authors

Presenter/Author Information

Yu Pei, Benjamin Hernandez, Dali Wang, Brian Smith, Scott Atchley, Fengming Yuan, and Chad Steed

International Environmental Modelling and Software Society (iEMSs)
8th International Congress on Environmental Modelling and Software
Toulouse, France, Sabine Sauvage, José-Miguel Sánchez-Pérez, Andrea Rizzoli (Eds.)
http://www.iemss.org/society/index.php/iemss-2016-proceedings

In situ Data Visualization of ACME Land Model Simulations

Yu Pei, Benjamin Hernandez, Dali Wang, Brian Smith, Scott Atchley, Fengming Yuan, Chad Steed
Oak Ridge National Laboratory,
Oak Ridge, TN USA
{peiy, hernandezarb,wangd, smithbe, atchleyes, yuanf, steedca}@ornl.gov

Abstract: With the ever increasing precision and complexity of various environmental models, writing large volumes of datpeiy@ornl.gova to disk for post-processing becomes a huge bottleneck, and simulation modules diagnostic and tracking become difficult for scientists as well. In this paper, we propose a loosely coupled in situ visualization framework to address these problems when working with a representative land model within the Accelerated Climate Modeling for Energy (ACME). ACME land model (ALM) simulates energy, water, carbon nitrogen fluxes and state variables for various land surfaces. Loosely coupled in situ framework means simulation and visualization communicate through network, while tightly coupled means they are integrated and run on the same nodes. Built with a high performance common communication interface (CCI) and a customized data visualization environment (DVE), we demonstrated our lightweight framework for a single site ALM simulation case running on Titan supercomputer. Our framework makes real-time simulation monitoring possible, and the customized DVE provides scientists an intuitive and useful way to explore results. We believe our flexible and performant in situ framework can be readily adapted to a wide range of simulations, expediting scientific discoveries. For future work, we will expand our current framework to include task scheduling, enabling multitask visualization of large-scale parallel environmental model simulations.

***Keywords**:* Accelerated Climate Modeling for Energy (ACME), ACME Land Model (ALM); In situ data visualization; common communication interface (CCI); High performance computing (HPC).

1 INTRODUCTION

Environmental modelling, such as, Earth system modelling, is a key approach toward better understanding of complex environmental systems. Improved Earth system model accuracy can help us discover potential energy sector vulnerabilities, forecast extreme weather events and understand human activities' impact on climate change. With the ever increasing computing capability in HPC, many high fidelity simulation models are constantly evolving to take advantage of the new system and improve model accuracy. But these complex Earth system models all face several challenges while in development. First, several teams of people (earth and computer scientists) are developing individual components within the Earth system model, making it difficult for scientists to track the whole system and understand the interaction between modules. Second, data storage speed lags far behind computing speed, creating huge bottleneck in disk I/O. The problem is compounded by the increased amount of data produced due to higher resolution simulation.

We are working on Accelerated Climate Modeling for Energy (ACME) and are facing the same problems. ACME model contains more than 350,000 lines of source code, more than 1,800 individual modules (Wang et al. 2014), and it can generate gigabyte of data per second for global simulation. Currently, scientists sometimes have to compromise for coarser time resolution results due to the large volumes of data. In this paper, we present our in situ data visualization framework for ALM addressing these difficulties.

Generally, in situ architectures can be classified into two categories: tightly coupled, where visualization/analysis and simulation run on the same nodes; Loosely coupled, where visualization/analysis and simulation run on concurrent resources and access data over network (Malakar et al. 2015, Rivi et al. 2012). Loosely coupled structure can provide the maximum amount of flexibility for each software component, less dependency between simulation and visualization development teams. The drawbacks are data will need to be transferred across node/network, duplicating data and having high latency. For tightly coupled structure, no data movement between simulation and visualization is required, but contending for computing resources, non-optimal visualization scheduling will impair simulation performance. What's more, careful co-design usually means close collaboration between simulation and visualization teams, incurring large amount of investment (Rivi et al. 2012, Wooding et al. 2016).

The main contributions of our work are:

1. Based on a flexible communication library, we built a loosely coupled in situ framework. The modular design makes each component replaceable, making the framework suitable for a broad range of simulations/cases while maintaining good performance.
2. We reduce the inherent complexity of visualizing ALM model datasets by designing an interactive data visualization environment that only analyses and present variables of interest.
3. Enabling environmental scientist to monitor variables interactively for function diagnostic, and to monitor model progression in real-time, an extension from previous work in Steed et al. (2013). More importantly, enabling the visualization/analysis of high resolution simulation results that wouldn't be possible before.

2. SYSTEM OVERVIEW

2.1 ACME Land Model (ALM)

Within the ACME model, ALM is the active component to simulate surface energy, water, carbon and nitrogen fluxes and state variables for both vegetated and non-vegetated land surfaces. The model represents several aspects of the land surface including surface heterogeneity and consists of submodels related to land biogeophysics, hydrological cycle, biogeochemistry and ecosystem dynamics (Wang et al. 2014).

Wang et al. (2014) investigated the call graph, internal hierarchical data structure for storing state variables, computation intensity and memory access pattern of community land model (CLM) – the predecessor of ALM model. The hierarchical data structure has not changed, but all of the state variables have been refactored into global variable space. This fact, and the static domain partitioning scheme are the key considerations when implementing code instrumentation and coordinating data transfer.

2.2 Common Communication Interface (CCI)

Targeted towards high performance computing environments as well as large data centers, CCI library provide a common Network Abstraction Layer (NAL) for persistent services as well as general interprocess communication (Atchley et al. 2011). What this means is that not only can it operate across different system architectures with good performance (utilizing underlying features such as zero-copy, operating system bypass etc), it also makes code coupling easier with minimal modification to the original programs. In our case, it provides a flexible abstraction for communicating between ALM and visualization program. CCI library is written in C, and can integrate easily with multiple programming languages.

2.3 Data Visualization Environment (DVE)

We designed a DVE to allow users to analyse interactively different aspects of the simulation in progress. The DVE has three main components, a communication module, a data visualization engine, and a graphical user interface (GUI).

The communication module opens a communication channel between the data visualization engine and the simulation, performs data unpacking and translates the simulation data model to the data visualization engine's data model. The data visualization engine generates and displays different types of plots and together with the GUI allows user to choose different carbon flux and nitrogen flux variables to plot their relationships, generate time series and visualize the evolution of these variables as the simulation advances.

2.4 Data Collection and Packaging

To monitor variables, we need to be able to capture data dependency and perform data packaging automatically. We adopted the recursive compiler-based code analyzer as in Yao et al. (2016) for this purpose. This allows the scientists to choose modules and variables that they are interested in for visualization.

3 SYSTEM IMPLEMENTATION

A tightly-coupled in situ visualization requires several teams' close collaboration; as noted by Kress et al. (2015), any change in the simulation or visualization code needs to be carefully managed. Instead, we built a loosely coupled framework with simpler and more flexible development, integration and deployment.

On the simulation side (Figure 1), there are two steps, data packaging and communication. We designed a set of user defined functions to package data and handle various communication related procedures, including endpoint creation, buffer allocation, message passing, remote memory access (RMA) and connection closing. Since CCI library and ALM were written in different programming languages (C and Fortran, respectively), we wrote the functions in C, and also wrote a Fortran wrapper to interface with ALM. With these utilities in place, we used the code analyzer described in section 2.4 to obtain a list of target variables, then used a Python script to automatically instrument those functions into the appropriate location in the source file.

The communication follows a client/server model, where the simulation is the client and the DVE is the server. First the server will initialize and emit an URI for clients connection. On the client side, the instrumented code will create one endpoint for each simulation process (internal job dispatching is coordinated with MPI), and each endpoint will read in the URI and connect with server. Currently we employ push-driven data transferring, namely client will pack the desired data for transfer once they have been generated, while server will keep polling on the endpoint for any incoming data from client then handle them accordingly.

On the visualization side (Figure. 1), the DVE provides the tools for in situ inspection of the simulation results. It was implemented in C++, in particular, the communication module uses the CCI library; the data visualization engine, uses the Visualization Toolkit (VTK 6.3) and the GUI was developed in Qt. Smooth user interaction with the system is enabled by running the communication module in an additional CPU thread.

DVE communication module works in a similar way as ACME's communication component, i.e. it unpacks received data, handles various communication related procedures, including endpoint creation and connection closing. Once the data, for a time step, are available in memory through RMA, are unpacked and converted to the visualization engine data model. In this case, the data model corresponds to the vtkTable data structure (Crossno et al. 2016). Each time step is then aggregated to a simulation vtkTable which stores all the incoming time steps.

The visualization engine reads the data from the simulation vtkTable, plots specific results according to the user input. VTK synchronizes the input with the resultant plot internally, updating the plot as new data are received.

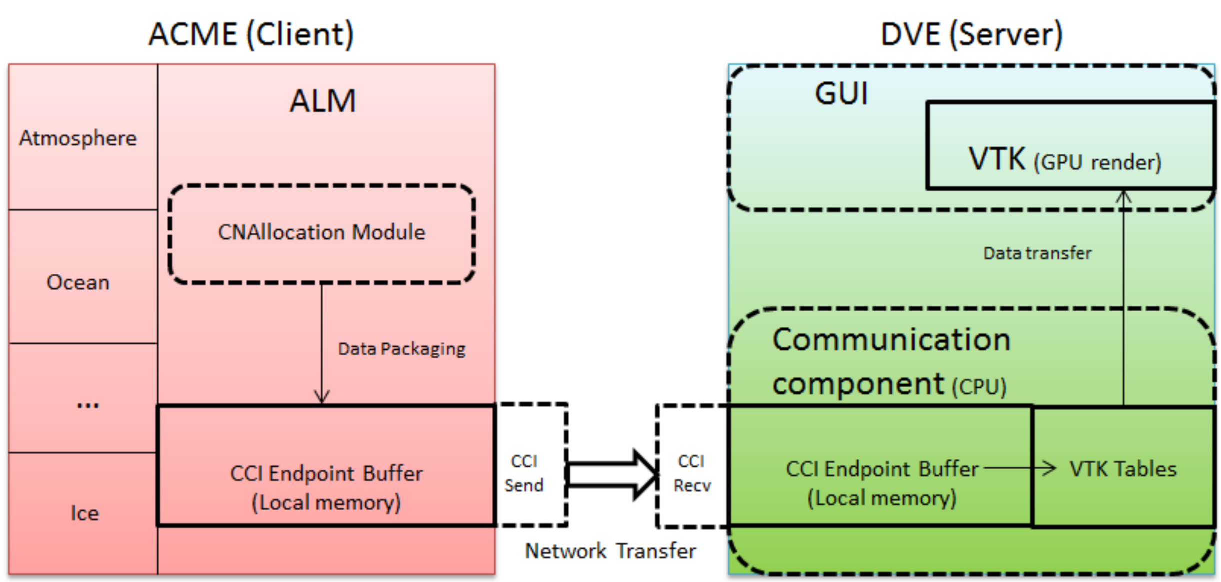

Figure 1: System architecture diagram. ACME has components to simulate atmosphere, ocean and coupling, but we highlights ALM with CCI data transfer. We show DVE's communication, rendering and GUI components.

4 DEMONSTRATION

To demonstrate our framework, we chose to instrument the CNAllocation module in ALM. This module allocates carbon and nitrogen in the terrestrial ecosystems. The experiment was run on Titan supercomputer at Oak Ridge National Laboratory (ORNL). Titan is a Cray XK7 system with 18,688 compute nodes, interconnected with Gemini network. Each node contain a 16-core AMD Opteron CPU and one NVIDIA Kepler GPU. Our study case is an offline ALM simulation with data atmosphere running on a single grid cell. We used one node for simulation and one node for DVE. After starting both processes, we can visualize the simulation progress using remote rendering. The result is shown in Figure. 2.

The model progress at half-hourly time step, simulating the climate at one Arctic location from year 1850. From the GUI, we can choose the interested variables to compare, plot against each other, or plot them as time series as model runs. The intuitive and interactive GUI enables scientists to start exploring the model runs immediately at the finest time resolution.

Although we didn't include formal performance measurements, the scientists in our group, who regularly post-process the simulation results, commented that our system provides a huge speed gain compared to writing results to disk then analyzing.

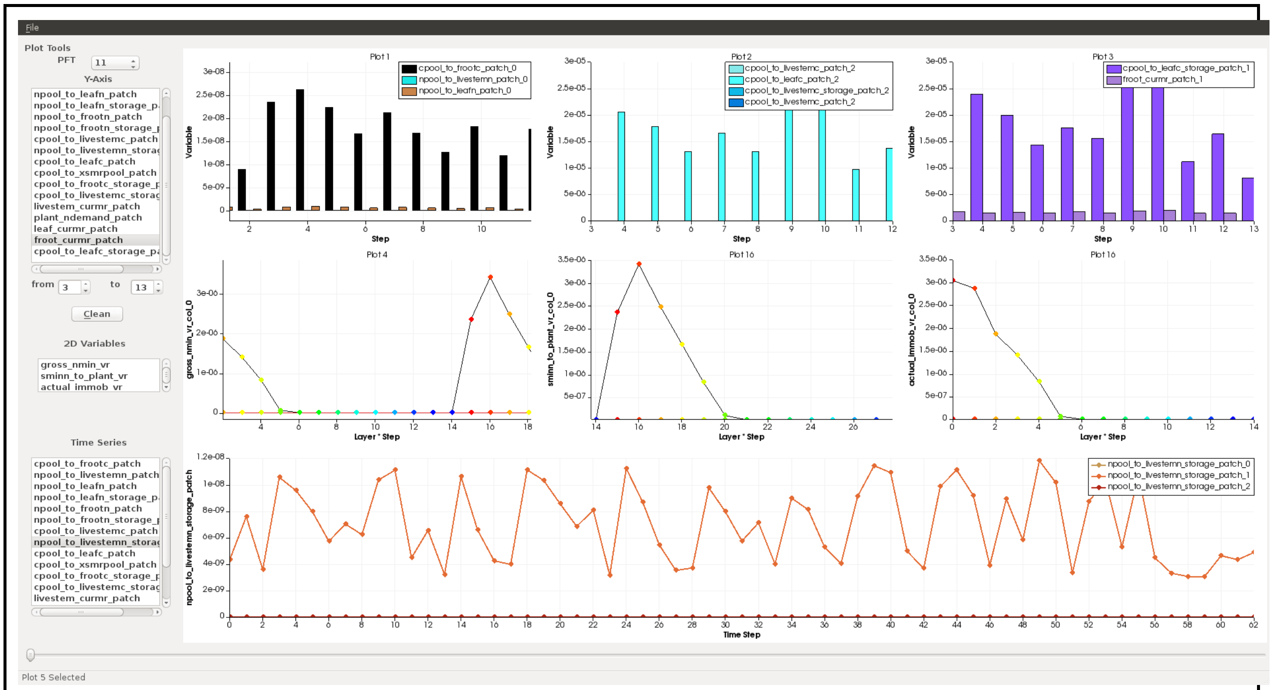

Figure 2: Visualization interface with various carbon and nitrogen variables. Top panels show the values for three plant types at a given time step. Bottom plot shows the time series of one variable. We can zoom in and out to view different time steps, drag to highlight certain time steps across all the plots.

5 RELATED WORK

As pointed out by Rivi et al. (2012), existing in situ data visualization frameworks includes Paraview with Catalyst (Yi et al. 2014) or ICARUS (Biddiscombe et al. 2011) plug-ins for communication and instrumentation, and VisIt with libsim library (Whitlock et al. 2011). Both Paraview and VisIt are popular open source application that support parallel, distributed visualization and analysis of large scale datasets. Further, both of them are built with VTK library for data model and many visualization algorithms. Paraview and Visit aim to provide a general solution for scientific visualization, although they offer a plugin extension mechanism, we opted to design and implement our own solution that fits ALM code, communication and visualization needs.

In situ visualization has been adopted in different domains, achieving results that would not be possible with traditional post-processing results on disk. Yu et al. (2010) applied this to large scale combustion simulation, capturing features would be missed with traditional restart files. Hernández et al. (2014) implemented in situ visualization for interactive crowd behavior simulation, significantly extending the experiment size while still providing real-time performance. In situ also enabled faster protein-folding analysis (Zhang et al., 2014).

Regarding in situ visualization in the climate modeling domain, to the best of our knowledge, Woodring et al (2016) is the first effort in this direction. The authors developed in situ eddy analysis in the Model for Prediction Across Scales (MPAS) ocean framework, achieving scalable analysis at large spatial grids and temporal frequencies that is not possible with the traditional climate post-processing. Both MPAS-ocean and ALM are part of ACME project, we are very glad that they have invested so much effort in improving the model. We are trying to achieve the same goal for ALM model, but not in a tightly coupled fashion, instead in a more nimble and loosely coupled framework.

6 CONCLUSION AND FUTURE WORK

In this paper, we proposed a loosely coupled in situ framework to monitor the complex ALM model and reduce I/O bottleneck in data post-processing when modeling Earth systems. We demonstrated the initial results running our framework for single site ALM simulation on Titan. With in situ visualization enabled, we can monitor simulation progress in real-time. Also it vastly reduced the time needed for writing results to disk then post-process, expediting knowledge discovery. The biggest contribution of our work is the construction of a loosely coupled in situ framework with CCI library and custom build visualization program. We believe that this flexible framework can provide a fast way for many large simulations to adopt in situ visualization/analytic for their own purposes.

For future work, we will extend our architecture shown in Figure 1 to support global scale simulations and multiple users, as a result, we will design new methods to handle and access big data scenarios efficiently. We also plan to enhance our DVE by statistical analysis tools and test the performance of our architecture at large-scale.

ACKNOWLEDGMENTS

This research was funded by the U.S. Department of Energy (DOE), Office of Science, Biological and Environmental Research (BER), Advanced Scientific Computing Research (ASCR), and an ORNL LDRD project (#7409). This research used resources of the Oak Ridge Leadership Computing Facility, located in the National Center for Computational Sciences at Oak Ridge National Laboratory (ORNL), which is managed by UT-Battelle LLC for the Department of Energy under contract DE-AC05-00OR22725.

REFERENCES

Atchley, S., Dillow, D., Shipman, G., Geoffray, P., Squyres, J.M., Bosilca, G., Minnich, R.. 2011. The Common Communication Interface (CCI). In *Proceedings of the 2011 IEEE 19th Annual Symposium on High Performance Interconnects* (HOTI '11). IEEE Computer Society, Washington, DC, USA, 51-60.

Biddiscombe, J., Soumagne, J., Oger, G., Guibert, D., Piccinali, J.G.. 2011. Parallel computational steering and analysis for HPC applications using a paraview interface and the HDF5 DSM virtual file driver. In *Proceedings of the 11th Eurographics conference on Parallel Graphics and Visualization* (EGPGV '11), Torsten Kuhlen, Renato Pajarola, and Kun Zhou (Eds.). Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 91-100.

Crossno, P., Moreland, K., Wilson, A. Wylie, B. vtkTable Class Reference. Kitware Inc. Access date March 31, 2016. http://www.vtk.org/doc/nightly/html/classvtkTable.html

Hernández, B., Pérez, H., Rudomin, I., Ruiz, S., de Gyves, O., Toledo, L.. 2014. Simulating and Visualizing Real-Time Crowds on GPU Clusters. *Computación y Sistemas*, *18*(4), 651-664

Kress, J., Klasky, S., Podhorszki, N., Choi, J., Childs, H., Pugmire, M. 2015. Loosely Coupled In Situ Visualization: A Perspective on Why It's Here to Stay. In Proceedings of the First Workshop on In Situ Infrastructures for Enabling Extreme-Scale Analysis and Visualization (ISAV2015). ACM, New York, NY, USA, 1-6.

Malakar, P., Vishwanath, V., Munson, T., Knight, C., Hereld, M., Leyffer, S., Papka, M.E. 2015. Optimal scheduling of in-situ analysis for large-scale scientific simulations. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis* (SC '15). ACM, New York, NY, USA, , Article 52 , 11 pages

Rivi, M., Calori, L., Muscianisi, G.. 2012. In-situ Visualization State-of-the-art and Some Use Cases. PRACE White Papers.

Steed, C.A., Ricciuto, D.M., Shipman, G., Smith, B., Thornton, P.E., Wang, D., Shi, X., Williams, D.N., Big data visual analytics for exploratory earth system simulation analysis, Computers & Geosciences, Volume 61, December 2013, Pages 71-82

Whitlock, B., Favre, J.M., Meredith, J.S.. 2011. Parallel in situ coupling of simulation with a fully featured visualization system. In *Proceedings of the 11th Eurographics conference on Parallel Graphics and Visualization* (EGPGV '11), Torsten Kuhlen, Renato Pajarola, and Kun Zhou (Eds.). Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 101-109.

Wang, D., Schuchart, J., Janjusic, T., Winkler, F., Xu, Y., Kartsaklis, C.. 2014. Toward Better Understanding of the Community Land Model within the Earth System Modeling Framework, Procedia Computer Science, Volume 29. Pages 1515-1524

Woodring, J., Petersen, M., Schmeisser, A., Patchett, J., Ahrens, J., Hagen, H.. 2016. In Situ Eddy Analysis in a High-Resolution Ocean Climate Model. In *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, no. 1, pp. 857-866.

Yu, H., Wang, C., Grout, R.W., Chen, J.H., Ma, K.L.. 2010. In Situ Visualization for Large-Scale Combustion Simulations. In *IEEE Computer Graphics and Applications*, vol. 30, no. 3, pp. 45-57.

Yi, H., Rasquin, M., Fang, J., Bolotnov, I.A.. 2014. In-situ visualization and computational steering for large-scale simulation of turbulent flows in complex geometries. *Big Data (Big Data), 2014 IEEE International Conference on*, Washington, DC, pp. 567-572.

Yao, Z., Jia, Y., Wang, D., Steed, C., Atchley, S.. 2016. In-situ Data Infrastructure for Scientific Unit Testing Platform. international Conference On Computational Science, {ICCS}. (To appear)

Zhang, B., Estrada, T., Cicotti, P., Taufer, M.. 2014. Enabling In-Situ Data Analysis for Large Protein-Folding Trajectory Datasets. In *Proceedings of the 2014 IEEE 28th International Parallel and Distributed Processing Symposium* (IPDPS '14). IEEE Computer Society, Washington, DC, USA, 221-230.