The PHASTA Science Gateway: Web-based Execution of Adaptive Computational Fluid Dynamics Simulations

Cameron W. Smith
Rensselaer Polytechnic Institute
110 8th Street
Troy, NY 12180
smithc11@rpi.edu

Eroma Abeysinghe
Indiana University
2709 East 10th Street
Bloomington, IN 47408
eabeysin@iu.edu

ABSTRACT

The Parallel Hierarchic Adaptive Stabilized Transient Analysis (PHASTA) software supports modeling compressible or incompressible, laminar or turbulent, steady or unsteady flows in 3D using unstructured grids. PHASTA, coupled with the Parallel Unstructured Mesh Infrastructure (PUMI), supports parallel, automated, adaptive simulation workflows. Researchers can easily execute these workflows on the TACC Stampede Xeon and Knights Landing nodes without being burdened by the details of each system using the PHASTA science gateway (created with Apache Airavata). In addition to abstracting away job execution and filesystem details, the gateway creates a searchable archive of past jobs to support reproducibility. Our poster presents the construction of the PHASTA gateway, the workflows it currently supports, and our ongoing efforts to expand functionality and the user base.

CCS CONCEPTS

• **Information systems** → **Distributed storage**; • **Computing methodologies** → **Massively parallel and high-performance simulations**; **Simulation tools**; • **Applied computing** → *Computer-aided design*; • **Software and its engineering** → *Software creation and management*; *Reusability*;

KEYWORDS

ACM proceedings, LaTeX, text tagging, Apache Airavata, Science Gateway

ACM Reference format:
Cameron W. Smith and Eroma Abeysinghe. 2017. The PHASTA Science Gateway: Web-based Execution of Adaptive Computational Fluid Dynamics Simulations. In *Proceedings of PEARC17, New Orleans, LA, USA, July 09-13, 2017,* 4 pages.
https://doi.org/10.1145/3093338.3104151

1 INTRODUCTION

The PHASTA Science Gateway enables execution and management of parallel computational fluid dynamics (CFD) simulations on high performance computing (HPC) systems [8] through a simple web-based user interface. CFD simulations use automated, adaptive workflows [14] for efficient execution of complex physical systems. These workflows use several software components (a reusable unit of composition with a set of interfaces to query and and modify encapsulated state information [17]) coupled with a in-memory streaming

PEARC17, July 09-13, 2017, New Orleans, LA, USA

ACM ISBN 978-1-4503-5272-7/17/07.
https://doi.org/10.1145/3093338.3104151

2 PHASTA ADAPTIVE WORKFLOWS

Workflows for simulating complex fluid flow phenomena leverage multiple existing software components to reduce workflow development and maintenance costs. PHASTA workflow components include:

- PHASTA: The **P**arallel **H**ierarchic **A**daptive **S**tabilized **T**ransient **A**nalysis software supports modeling compressible or incompressible, laminar or turbulent, steady or unsteady flows in 3D, using unstructured meshes [1, 9, 10, 15, 18]. The PHASTA Science Gateway enables researchers to easily execute adaptive workflows on the TACC Stampede and RPI Blue Gene/Q systems.
- Simmetrix MeshSim and GeomSim: Simmetrix MeshSim provides automatic conformal, unstructured mesh generation for Parasolid, ACIS, and discrete CAD models [13]. A portable, parametric geometric model representation is provided by the GeomSim modeling kernel.
- PUMI: **P**arallel **U**nstructured **M**esh **I**nfrastructure provides mesh adaptation, partitioning, and field management services [4].
- ParMA: **Par**titioning **U**sing **M**esh **A**djacencies provides scalable dynamic load balancing [16]. For PHASTA it specifically targets the mesh entities holding degrees of freedom. The balance of these entities is proportional to the scalability of the computationally dominant linear algebra work in the analysis procedures [16, 20].

Using these components, PHASTA workflows have been developed to study active flow control for aerodynamics (planes, engine inlets, and wind turbines) [8], multi-phase bubbly flows (nuclear reactor performance and accident scenarios) [1], and biomedical flows (cardiovascular and respiratory flows) [5].

3 RUNNING A PHASTA SIMULATION

The steps for defining and executing a PHASTA adaptive simulation are:

(1) Mesh the geometric model
(2) Define boundary and initial conditions
(3) Set execution control parameters
(4) Upload files to the target computational system

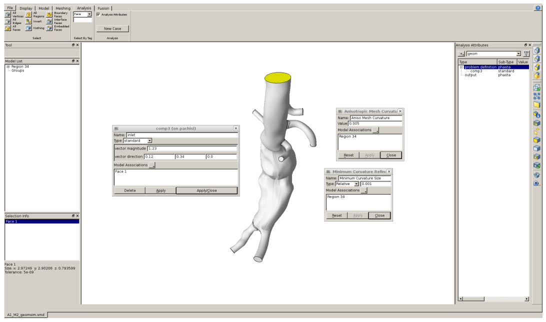

Figure 1: Simmetrix SimModeler mesh generation and problem definition for an abdominal aortic aneurysm (AAA) case.

(5) Specify the needed computational resources
(6) Submit simulation
(7) Download results when execution completes
(8) Postprocess the results

User's complete steps 1- 3 using the Simmetrix SimModeler [12] graphical user interface depicted in Figure 1. Alternatively, but with relatively limited functionality, GMSH [2, 3] and plain text files can be used to generate the mesh, and specify the problem definition information. In either case, problem definition and mesh generation controls are specified with respect to a geometric model [11].

Once the problem is defined, the necessary input files are uploaded to the target HPC system, the simulation is executed, and the results downloaded for subsequent post processing. These steps, 4 through 7, are typically completed using command-line tools whose syntax can vary dramatically between HPC systems. For example, submitting a job with MOAB, PBS, and SLURM job schedulers on different systems requires specific incantations based on knowledge of site-specific installation defaults, the job queues, and project allocations. The PHASTA Science Gateway provides a single simplified interface for completing steps 4 through 7 on multiple HPC systems and managing generated simulation data.

4 PHASTA GATEWAY & APACHE AIRAVATA

4.1 PHASTA Gateway with Apache Airavata

The PHASTA science gateway uses **Sci**ence **Ga**teway **P**latform as a Service [7] (SciGap). The SciGaP platform provides gateway services via Apache Airavata [6]. Airavata core infrastructure capabilities required by the PHASTA gateway include: user identity, accounts, authorization, and the ability to access multiple computational resources. Figure 2 depicts the components of the SciGaP platform and its functional interactions with a gateway instance and computing resources. Note, the hosted SciGaP platform is multi-tenanted and manages multiple science gateways.

4.1.1 User Accounts, Authentication & Authorization. In order to provide gateway access to users approved by the gateway administrators, the PHASTA gateway provides authentication and authorization services. With gateway administrator approval, users can access and use the gateway and underlying computational resources. Gateway users will be given one of the four available

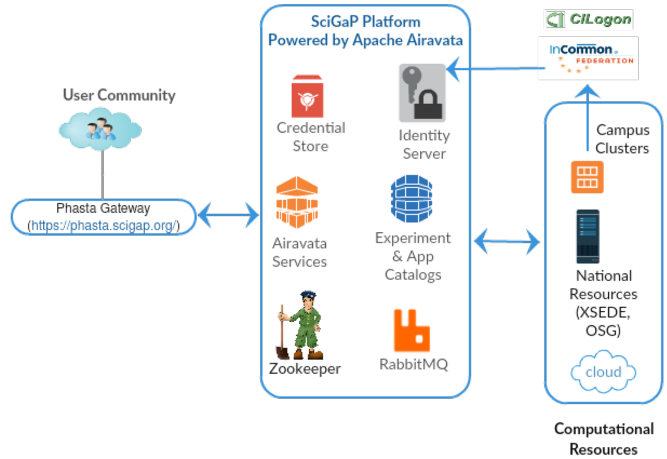

Figure 2: PHASTA Gateway with Apache Airavata.

roles and associated permissions. The current user roles are: 'User-Pending', 'Gateway-User', 'Admin-Read-Only' and 'Admin'. The 'User-Pending' role cannot use any Airavata services provided by the gateway; it is given to users pending approval from gateway administrators. The 'Gateway-User' can use Airavata services for creating, monitoring, sharing and cloning experiments and also add own compute resource allocation details into the gateway and use them in job submission to clusters. The users with 'Admin' role can control metadata for accessing computing resources and applications, manage users, and monitor and access all user experiments information. 'Admin-Read-Only' can access all 'Admin' related information but cannot modify it. For authentication and authorization Apache Airavata uses the third party identity management service from WSO2 (wso2.com/identity-and-access-management).

4.1.2 Computational Experiments. The PHASTA gateway allows users and gateway administrators to create, execute, monitor, share, and manage computational experiments. Figure 3 depicts the PHASTA interface for experiment creation. Using this interface gateway users can upload gateway specified input data and later download output data from completed experiments. When the application archive feature is enabled all experiment level data will be archived and made available for the users via the gateway interface. When experiments are shared with each other, users can also use other gateway users experimental work to execute computational experiments. As part of managing their experiments users can cancel running experiments and also clone existing and execute new experiments on compute resources/clusters.

4.1.3 Monitoring Progress. During experiment creation users can provide their email to receive messages at job start and end. Additionally, once the experiment is launched, the experiment summary interface is automatically refreshed in order to show the real time status of the job submitted into the compute resource/cluster. Figure 4 depicts this interface. Regular users can monitor experiments owned by them and shared with them by other gateway users. Gateway administrators can monitor all gateway experiments using the Experiment Statistics page in the Admin Dashboard.

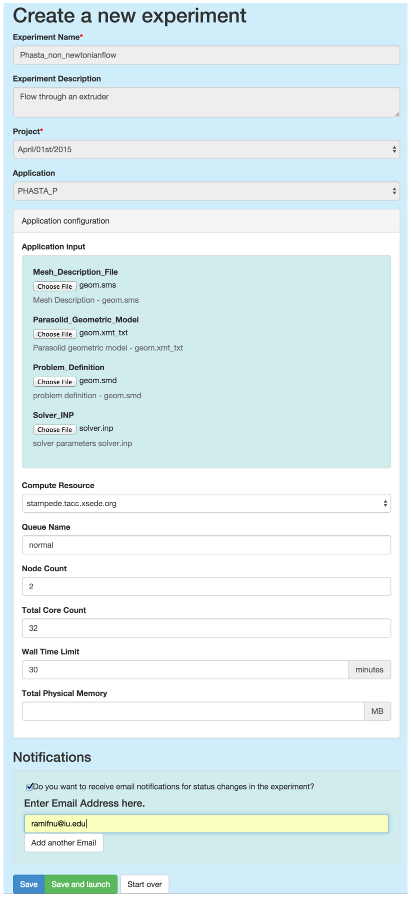

Figure 3: PHASTA gateway experiment creation interface.

Figure 4: PHASTA gateway experiment summary interface.

4.1.4 *Admin Dashboard.*

The Admin Dashboard is the workspace for the gateway administrator within the gateway. Through it, all the administrator features previously discussed are controlled. Additionally, administrators can create notifications for users to provide usage tips, functionality updates, or maintenance outages.

5 ADAPTIVE DAMBREAK WORKFLOW

The PHASTA gateway enables execution of adaptive workflows supporting multiple applications. One such application is depicted in Figure 5; the evolution of the adaptive mesh for a dam-break test case ran on ALCF Theta using two-phase incompressible PHASTA-chef with data streams [9, 14]. In the dam-break simulation the dense fluid (water) is initially held against the left wall (not pictured) in a square column created by a fictitious constraint representing a dam. The remainder of the domain is air. When the constraint is removed, as if the dam broke, the dense fluid falls and advances to the right [9]. A distance and curvature-based refinement band tracks the air-water interface. Outside of these bands the mesh is graded to a reference coarse size.

The two-phase adaptive workflow starts by loading the PUMI partitioned mesh, geometric model, and problem definition information. Next, the chef preprocessor executes adjacency-based mesh entity reordering [19] to improve the efficiency of the assembly and linear algebra solution procedures. Then, chef creates the finite element mesh (i.e., nodes and element connectivity), solution field, and structures containing the point-to-point communication protocols and boundary conditions and writes it to streams [14].

After the initial execution of chef, the solve-adapt cycle runs until the requested number of solver time steps is reached. The PHASTA solver first reads its input information from chef via files or streams, then executes an analyst-specified number of time steps, and computes the distance-based mesh size field. The solver then writes the computed mesh size field and solution field to streams. Those fields are read and attached to the PUMI mesh. Next, chef drives MeshAdapt with the mesh size field. To prevent memory exhaustion during mesh refinement procedures, ParMETIS part k-way graph re-partitioning (via Zoltan) is called using weights that approximate the change in mesh element count on each part. After adaptation, chef executes preprocessing as previously described.

6 CONCLUSIONS

The PHASTA Science Gateway supports execution of parallel, adaptive, in-memory workflows for aerospace, nuclear, and biomedical applications on remote high performance computing systems without burdening the users with system specific details while providing data management capabilities. Ongoing efforts with the Airavata team to improve the PHASTA gateway include the support for large file transfers via Globus, and in-situ post-processing and visualization with ParaView. These features have been identified by PHASTA users as critical needs and should drive expansion of the user base.

In addition to specific improvements targeting the PHASTA gateway, the Airavata team is working on multiple functionalities to improve the experience for gateway users and administrators. Gateway users will benefit from better authentication and authorization

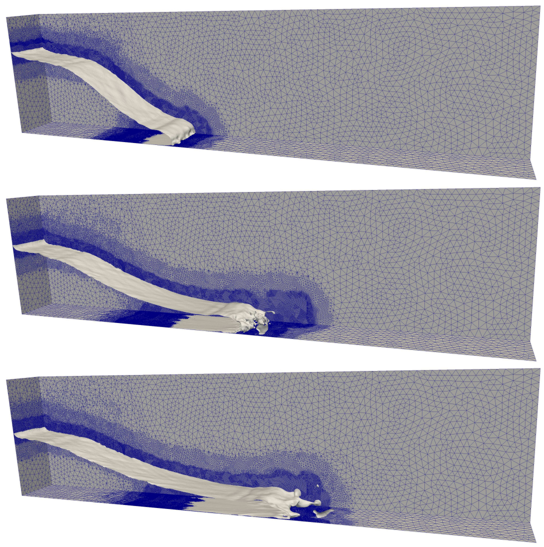

Figure 5: Evolution of an adaptive dam-break case ran on 2048 processes of the ALCF Theta system using two-phase, incompressible PHASTA coupled to PUMI unstructured mesh adaptation with data streams. Each image (top to bottom) represents an advancement in physical time by 2/100 of a second. The air-water phasic interface iso-surface is shown in gray.

methods (such as CILogon (www.cilogon.org)), and accurate automatic selection of resources for a particular job submission. Gateway administrators will get more control over compute and storage resource registration, and will have automated mechanisms for the addition of public keys generated by the gateway to authorized keys file in individual user accounts in clusters.

ACKNOWLEDGMENTS

The authors would like to thank Professor Kenneth Jansen at the University of Colorado Boulder for supporting development of PHASTA adaptive workflows, and the Science Gateway team at Indiana: Sudhakar Pamidighantam, Suresh Marru and Marlon Pierce.

Portions of this research was supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, under award DE-SC00066117 (FASTMath SciDAC Institute).

This work used the Extreme Science and Engineering Discovery Environment (XSEDE), which is supported by National Science Foundation grant number ACI-1548562.

We thank Lars Koesterke for his assistance with use of the Stampede Knights Landing nodes, which was made possible through the XSEDE Extended Collaborative Support Service (ECSS) program.

REFERENCES

[1] Azat Yu. Galimov, Onkar Sahni, Richard T. Lahey Jr., Mark S. Shephard, Donald A. Drew, and Kenneth E. Jansen. 2010. Parallel Adaptive Simulation of a Plunging Liquid Jet. *Acta Mathematica Scientia* 30, 2 (March 2010), 522–538. https://doi.org/10.1016/S0252-9602(10)60060-4

[2] Christophe Geuzaine and Jean-François Remacle. 2009. Gmsh: A 3-D finite element mesh generator with built-in pre- and post-processing facilities. *Int. J. Numerical Methods in Eng.* 79, 11 (May 2009), 1309–1331. https://doi.org/10.1002/nme.2579

[3] Christophe Geuzaine and Jean-François Remacle. 2013. Gmsh: A three-dimensional finite element mesh generator with built-in pre- and post-processing facilities. (2013). http://gmsh.info/

[4] Daniel A. Ibanez, E. Seegyoung Seol, Cameron W. Smith, and Mark S. Shephard. 2016. PUMI: Parallel Unstructured Mesh Infrastructure. *ACM Trans. Math. Softw.* 42, 3, Article 17 (May 2016), 28 pages. https://doi.org/10.1145/2814935

[5] Victor L Marrero, John A Tichy, Onkar Sahni, and Kenneth E Jansen. 2014. Numerical study of purely viscous non-Newtonian flow in an abdominal aortic aneurysm. *J. Biomechanical Eng.* 136, 10 (Oct. 2014), 101001–1–101001–10. https://doi.org/10.1115/1.4027488

[6] Suresh Marru, Lahiru Gunathilake, Chathura Herath, Patanachai Tangchaisin, Marlon Pierce, Chris Mattmann, Raminder Singh, Thilina Gunarathne, Eran Chinthaka, Ross Gardler, et al. 2011. Apache airavata: a framework for distributed applications and computational workflows. In *Proceedings of the 2011 ACM workshop on Gateway computing environments*. ACM, 21–28.

[7] Marlon Pierce, Suresh Marru, Mark A Miller, Amit Majumdar, and Borries Demeler. 2013. Science Gateway Operational Sustainability: Adopting a PlatformasaService Approach. (2013).

[8] Michel Rasquin, Cameron Smith, Kedar Chitale, E. Seegyoung Seol, Benjamin A. Matthews, Jeffrey L. Martin, Onkar Sahni, Raymond M. Loy, Mark S. Shephard, and Kenneth E. Jansen. 2014. Scalable Implicit Flow Solver for Realistic Wing Simulations with Flow Control. *Comput. in Sci. & Eng.* 16, 6 (Dec. 2014), 13–21. https://doi.org/10.1109/MCSE.2014.75

[9] Joseph M. Rodriguez, Onkar Sahni, Richard T. Lahey Jr., and Kenneth E. Jansen. 2013. A Parallel Adaptive Mesh Method for the Numerical Simulation of Multiphase Flows. *Comput. & Fluids* 87 (Oct. 2013), 115–131. https://doi.org/10.1016/j.compfluid.2013.04.004

[10] Onkar Sahni, Min Zhou, Mark S Shephard, and Kenneth E Jansen. 2009. Scalable implicit finite element solver for massively parallel processing with demonstration to 160k cores. In *Proc. Int. Conf. High Performance Comput., Networking, Storage and Anal. (SC)*. IEEE, 1–12.

[11] Mark S Shephard, M.W. Beall, R.M. O'Bara, and B.E. Webster. 2004. Toward simulation-based design. *Finite Elements in Anal. and Design* 40, 12 (July 2004), 1575–1598.

[12] Simmetrix. 1997. Simmetrix: Enabling Simulation-Based Design. (1997). http://www.simmetrix.com/

[13] Simmetrix Inc. 2017. *Simulation Modeling Suite 11.0 Documentation*. Simmetrix Inc., Clifton Park, NY.

[14] Cameron W. Smith, Brian Granzow, Gerrett Diamond, Dan A. Ibanez, Onkar Sahni, Kenneth E. Jansen, and Mark S. Shephard. 2017. In-memory Integration of Existing Software Components for Parallel Adaptive Unstructured Mesh Workflows. *Concurrency and Computation: Practice and Experience* (April 2017), 1–19. submitted for publication.

[15] Cameron W. Smith and Kenneth E. Jansen. 2015. PHASTA GitHub repository. (2015). https://github.com/PHASTA/phasta

[16] Cameron W. Smith, Michel Rasquin, Dan Ibanez, Kenneth E. Jansen, and Mark S. Shephard. 2017. Improving Unstructured Mesh Partitions for Multiple Criteria Using Mesh Adjacencies. *SIAM J. Scientific Comput.* (Feb. 2017), 1–25. submitted for publication.

[17] Clemens Szyperski. 2002. *Component Software: Beyond Object-Oriented Programming* (second ed.). ACM Press and Addison-Wesley, New York, NY, USA.

[18] M. Zhou, O. Sahni, H. J. Kim, C. A. Figueroa, C. A. Taylor, M. S. Shephard, and K. E. Jansen. 2010. Cardiovascular Flow Simulation at Extreme Scale. *Computational Mech.* 46, 1 (Dec. 2010), 71–82.

[19] Min Zhou, Onkar Sahni, Mark S Shephard, Christopher D Carothers, and Kenneth E Jansen. 2010. Adjacency-based data reordering algorithm for acceleration of finite element computations. *Scientific Programming* 18, 2 (May 2010), 107–123.

[20] Min Zhou, Onkar Sahni, Ting Xie, Mark S Shephard, and Kenneth E Jansen. 2012. Unstructured mesh partition improvement for implicit finite element at extreme scale. *The J. Supercomputing* 59, 3 (Dec. 2012), 1218–1228.