# Middleware for Data Intensive Analytics on HPC

Ioannis Paraskevakos
Rutgers, The State University of New Jersey
Piscataway, New Jersey, USA
i.paraskev@rutgers.edu

**ACM Reference Format:**
Ioannis Paraskevakos. 2018. Middleware for Data Intensive Analytics on HPC. In *Proceedings of 47th International Conference on Parallel Processing, August 13-16, 2018, University of Oregon, Eugene, Oregon, USA (ICPP).* ACM, New York, NY, USA, 2 pages. https://doi.org/

## 1 EXTENDED ABSTRACT

Many scientific applications have immense amounts of data which require analysis. The execution time of the analysis highly depends on data volume. In addition, based on the application, the analysis may require execution either using same or different resources as the simulations. Thus, there is a high demand for an efficient and scalable solution which provides resource management and workflow abstractions for analyzing data on distributed resources, like High Performance Computing (HPC) resources.

### 1.1 Scientific Drivers

Molecular Dynamics (MD) simulations are significant consumers of supercomputing cycles, producing immense amounts of data. A typical $\mu sec$ MD simulation of physical system of $O(100k)$ atoms can produce from $O(10)$ to $O(1000)$ GBs of data [2]. In addition to being the prototypical HPC application, there is increasingly a need for the analysis to be integrated with simulations and drive the next stages of execution (analysis-driven- simulation) [1]. The analysis phase must be performed quickly and efficiently in order to steer the simulations.

Geo-sciences acquire satellite imagery data for understanding biological, hydrological and geological functioning of the polar regions. In addition, efficient algorithms that analyze high-resolution images is available. Although, as the data sizes for analyzing large-scale regions, like the Antarctic, increase, the computational requirements are becoming a limitation. As a result, it becomes necessary to use efficient and parallel frameworks to analyze and classify high-resolution satellite imagery using distributed resources.

### 1.2 Current & Future Challenges

There are three major challenges that this research tries to address:

1. Provide abstractions that capture common scientific analysis patterns and be science domain independent. A preliminary comparison between the data analysis requirements and patterns of the use cases in 1.1 suggests that they are fairly similar.
2. Provide a set of extensible and pluggable abstractions that operate on distinct layers from the application layer down to the resource interface. Scientific use cases evolve constantly. Having distinct layers of abstractions that are extensible and pluggable will allow the middleware to evolve without having to engineer a full stack solution.
3. Provide APIs that are not software stack specific. There is, already, a plethora of frameworks and software stacks (Apache Stack, PyData) that support data intensive applications [3]. The provided APIs should be able to interface with frameworks that provide the required features seamlessly and with the least possible engineering effort.

### 1.3 Proposed Approach

To address the challenges in 1.1 we propose to use the Building Blocks [8] approach. Each block is characterized by four design principles: (1) Self-sufficiency, (2) Interoperability, (3) Composability, and (4) Extensibility

In addition, the Building Block approach identifies four functional levels: **Level 4. Application Description**: Describes the requirements and semantics of the application, **Level 3. Workload Management System:** Expresses the applications as workloads, **Level 2. Task Runtime System:** describes how the tasks of the workload are executed, and **Level 1. Resources**

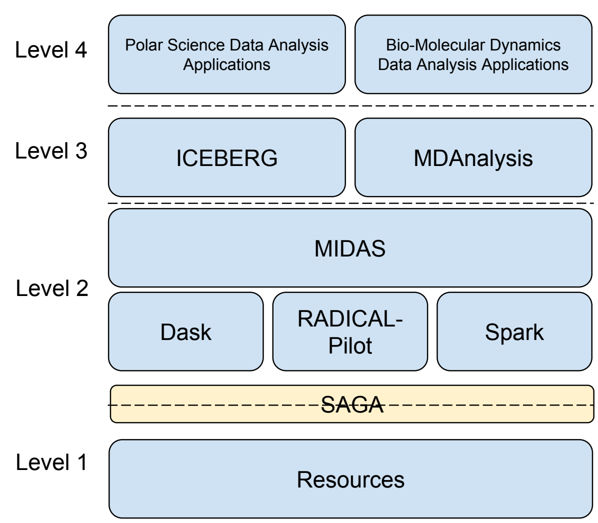


**Figure 1: Proposed approach four functional layers, along with their respective building blocks**


ACM ISBN .
https://doi.org/

Figure 1 shows graphically the proposed approach. Each layer has one or more building blocks. Level 1 is comprised by computing resources such as High Perfomance Computing resources (HPCs), Clouds and Grids. Level 2 shows the building blocks that describe how the tasks will be executed. Dask [7], RADICAL-Pilot [4], and Spark [9] are task execution engines. MIDAS stands for MIddleware for Data-intensive Applications and Science. MIDAS supports data-intensive analysis applications in conjuction with traditional HPC applications.

Level 3 shows the building blocks that express the applications as workloads. ICEBERG will provide the building block to describe and support image analytics for geo-sciences. MDAnalysis [5, 6] provides several algorithms for MD simulation data analysis. Some of these capabilities will be extended by using MIDAS. Level 4 has the application descriptions.

## 1.4 Preliminery results

MIDAS has been used for two distinct application and analysis: First, it supports data-intensive analysis applications in conjunction with traditional HPC applications. It has enabled the analysis of biophysical simulations systems that were not feasible before. Second, MIDAS is being used to analyze high-resolution satellite imagery and derived products (such as digital elevation models). In addition to large volumes of data involved, the analysis of images is computationally intensive and requires high-performance and distributed resources.

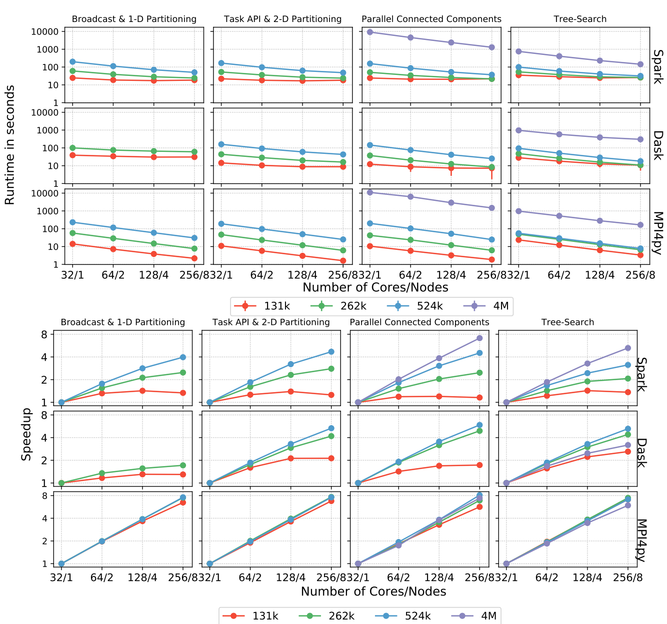


**Figure 2: Bio-Molecular Dynamics Use case: Leaflet finder performance using some capabilities offered by MIDAS**

A bio-molecular dynamics application, called Leaflet Finder [5], was implemented via the capabilities MIDAS offers. The Leaflet Finder is a graph-based algorithm to detect continuous lipid membrane leaflets in a MD simulation. Figure 2 shows the characterization of those capabilities.

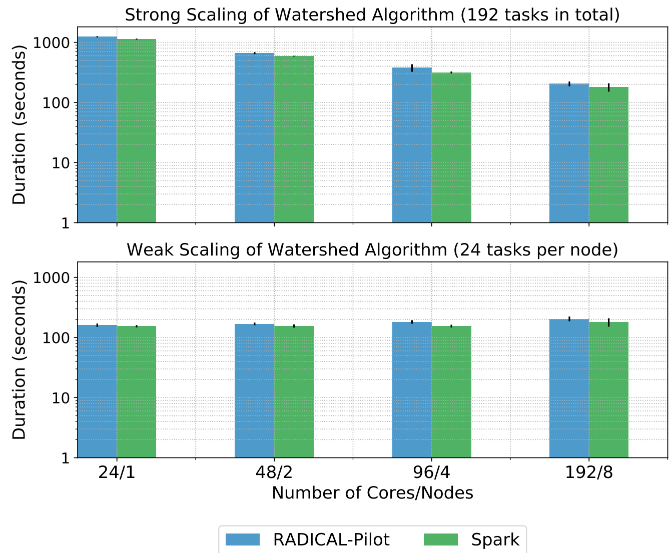


**Figure 3: Polar Imagery Use Case: Performance of Watershed algorithm using MIDAS capabilities**

In addition, an image analysis algorithm was implemented using capabilities that MIDAS provides. This algorithm was characterized using data from polar sciences. Figure 3 shows the performance characterization of this algorithm in strong and weak scaling.

*Acknowledgements* This work is funded by NSF 1443054 and 1440677. Computational resources were provided by NSF XRAC award TG-MCB090174. The NSF Office of Advanced Cyberinfrastructure, Registration Awards to ICPP 2018 PhD Forum for Student participants.

## REFERENCES


[1] V. Balasubramanian, I. Bethune, A. Shkurti, E. Breitmoser, E. Hruska, C. Clementi, C. Laughton, and S. Jha. 2016. ExTASY: Scalable and flexible coupling of MD simulations and advanced sampling techniques. In *2016 IEEE 12th International Conference on e-Science (e-Science)*. 361–370.

[2] T. Cheatham and D. Roe. 2015. The impact of heterogeneous computing on workflows for biomolecular simulation and analysis. *Computing in Science Engineering* 17, 2 (2015), 30–39.

[3] Shantenu Jha, Judy Qiu, André Luckow, Pradeep Kumar Mantha, and Geoffrey Charles Fox. 2014. A Tale of Two Data-Intensive Paradigms: Applications, Abstractions, and Architectures. *Proceedings of 3rd IEEE Internation Congress of Big Data* abs/1403.1528 (2014).

[4] Andre Merzky, Matteo Turilli, Manuel Maldonado, Mark Santcroos, and Shantenu Jha. 2018. Using Pilot Systems to Execute Many Task Workloads on Supercomputers. (2018). http://arxiv.org/abs/1512.08194.

[5] Naveen Michaud-Agrawal, Elizabeth J. Denning, Thomas B. Woolf, and Oliver Beckstein. 2011. MDAnalysis: A toolkit for the analysis of molecular dynamics simulations. *Journal of Computational Chemistry* 32, 10 (2011), 2319–2327.

[6] Richard J. Gowers, Max Linke, Jonathan Barnoud, Tyler J. E. Reddy, Manuel N. Melo, Sean L. Seyler, Jan Domański, David L. Dotson, Sébastien Buchoux, Ian M. Kenney, and Oliver Beckstein. 2016. MDAnalysis: A Python Package for the Rapid Analysis of Molecular Dynamics Simulations. In *Proceedings of the 15th Python in Science Conference*, Sebastian Benthall and Scott Rostrup (Eds.). 98 – 105.

[7] Matthew Rocklin. 2015. Dask: Parallel Computation with Blocked algorithms and Task Scheduling. In *Proceedings of the 14th Python in Science Conference*, Kathryn Huff and James Bergstra (Eds.). 130 – 136.

[8] Matteo Turilli, Andre Merzky, Vivek Balasubramanian, and Shantenu Jha. 2018. A Building Blocks Approach towards Domain Specific Workflow Systems? *Short Paper (IEEE/ACM CCGrid 2018)* (2018). http://arxiv.org/abs/1609.03484.

[9] Matei Zaharia, Mosharaf Chowdhury, Michael J. Franklin, Scott Shenker, and Ion Stoica. 2010. Spark: Cluster Computing with Working Sets. In *Proceedings of the 2Nd USENIX Conference on Hot Topics in Cloud Computing (HotCloud'10)*. USENIX Association, Berkeley, CA, USA, 10–10.