Dynamic Resource Management for Elastic Scientific Workflows using PMIx/PRRTE

Rajat Bhattarai, Tennessee Tech University

rbhattara43@tntech.edu

September 27, 2024

Challenges for Elastic Workflows

HPC Resource Managers (RMs) not suitable for workflows, let alone elastic

- Job abstraction rather than workflow-aware abstraction
- static resource allocation model
- Optimized for a few, long running applications

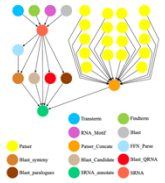
Computational needs of tasks in workflows might not be known when submitted

- Multiple stages with different requirements
- Interesting phenomenon may occur
- Current Approach: request maximum resources potentially wasting resources

Elasticity in only some Workflow Managers like Parsl [1], not fully efficient

- Resources as block (jobs)
- Big job allocations may have long waiting time
- No finer control on elasticity
- Smaller blocks may help but
  - Spanning same HPC applications across multiple jobs can get complicated
  - User limit on simultaneous jobs impacts, e.g., ALCF's Polaris allows by request only; max 100 jobs running/ accruing/ queued per-project

CyberShake Workflow


SIPHT Bioinformatics Workflow


Montage Workflow

Epigenomics Workflow

Common Workflow DAGs, Adapted from Juve, G., Chervenak, A., Deelman, E., Bharathi, S., Mehta, G., & Vahi, K. (2013). Characterizing and profiling scientific workflows. Future generation computer systems, 29(3), 682-692.

Can Process Management Interface for Exascale (PMIx) [2] help?

- Mediator between applications and RMs enabling implementation-independent interactions like flexible resource allocation
  - *PMIx_Allocation_request* API
  - Expanding and shrinking Distributed Virtual Machine (DVMs) in PMIx Reference Runtime Environment (PRRTE)
- MPI and PMIx standards advancing to support malleability features using PMIx [7]
  - e.g., MPI Sessions, ParaStation MPI, Dynamic Processes With Process Sets
  - Structure malleability into multiple layers, with clearly defined tasks and services for each layer based on the PMIx

Approach: Enhance Parsl to support workflow elasticity and develop a dynamic resource manager atop Slurm to create a Dynamic Resource Management Model for HPC platforms with finer granularity, both leveraging PMIx.

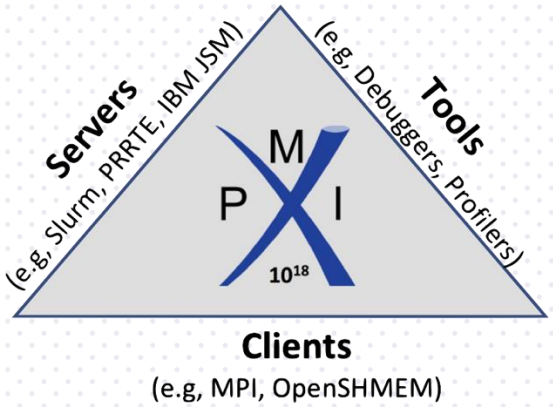
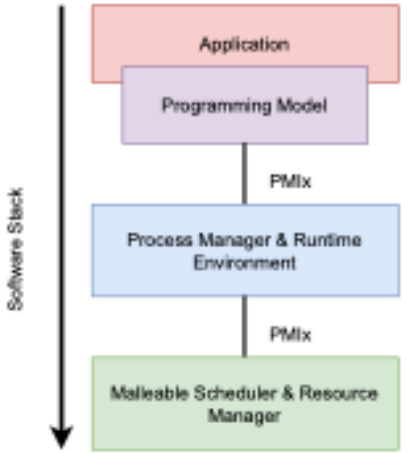


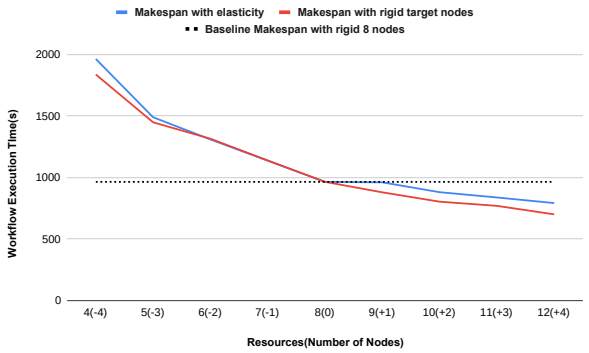
Figure from SC22 BoF: Charting the PMIx Roadmap



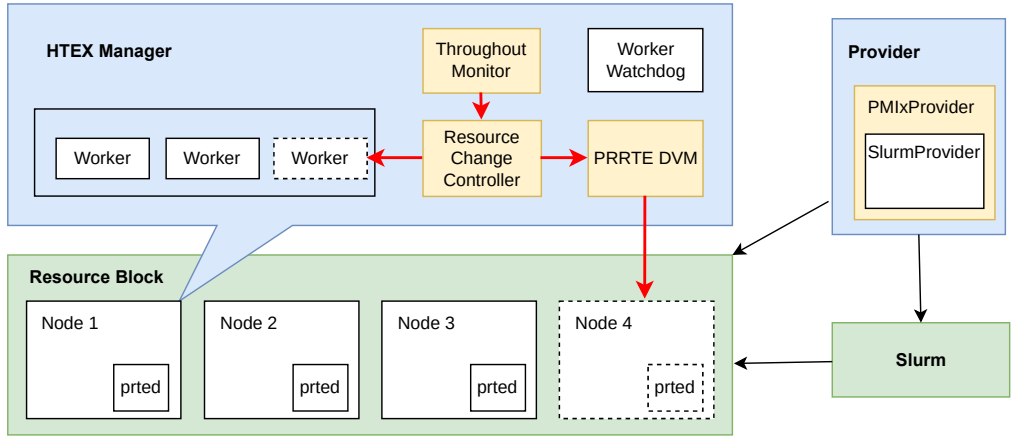
A multi-layer model of the system software on HPC systems subject to mastackllleability [7]

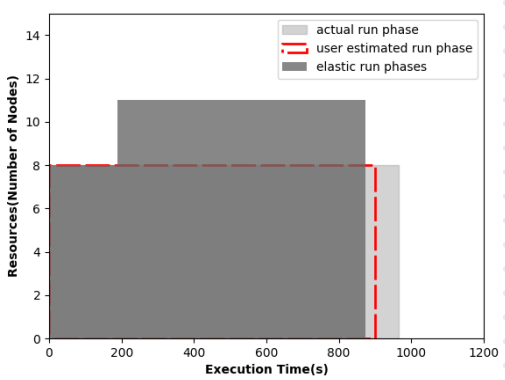
PMIx-Enabled Elasticity: Within Managers

- Add or remove workers and nodes [3]
  - PMIx-based Provider
  - HTEX Manager initiating and executing resource changes either at user specified time or during different performance-based events like overprovisioning and underprovisioning
- Suitable when workers can spawn workflow task in remote nodes (shell out), like, @bash_app with prun, srun or mpirun
  - Difficult for workflow containing @python_app
  - How to launch just workers that can run python function from a manager located at one node to another node?

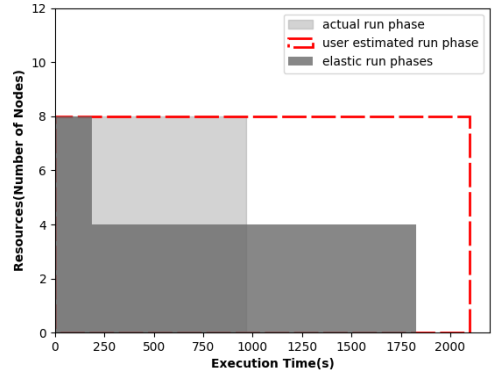


Blast Workflow Results from [3]





Under Provisioning Case

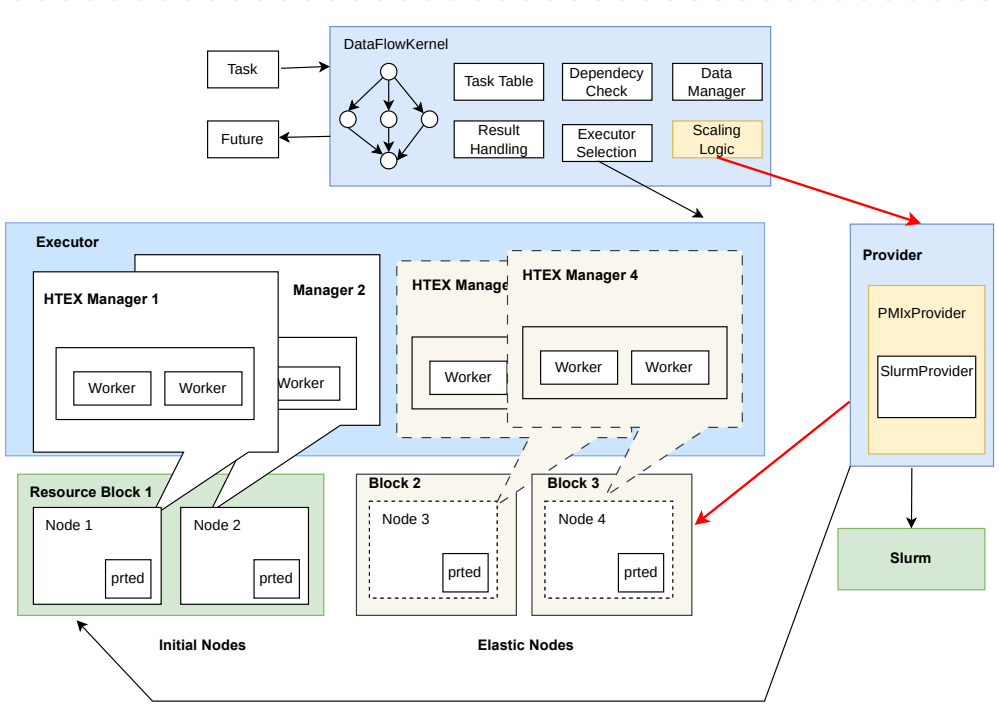


Over Provisioning Case

Modest speedup on expansion and slowdown on shrinkage of nodes. Can Correct Under Provisioning and Over Provisioning.

PMIx-Enabled Elasticity: Manager Level

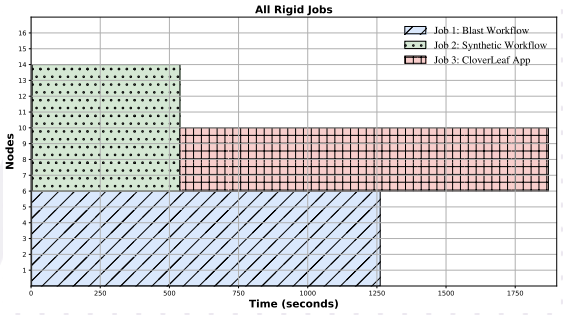
- Adding or removing managers
- Every newly added resource has manager with workers bind to cores
- Applicable to workflow containing python apps
- Like Block elasticity of Parsl, instead of duplicating block for expansion, PMIxProvider can provide resource blocks in granular level where managers can be placed

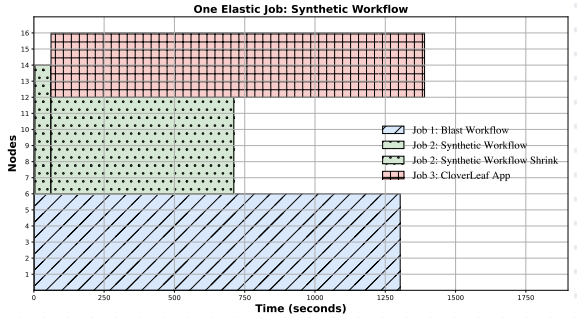


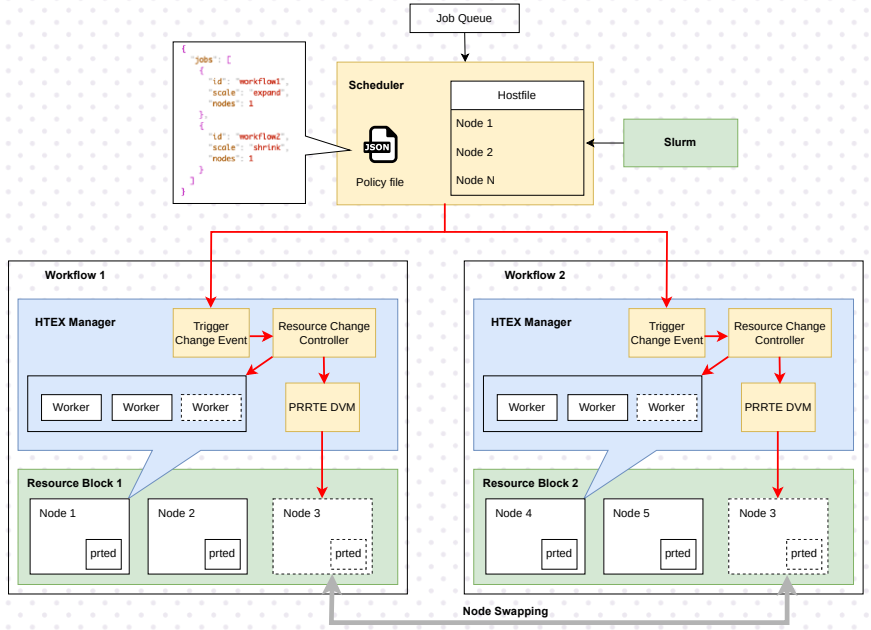
PMIx Based Manager Level Elasticity in Parsl

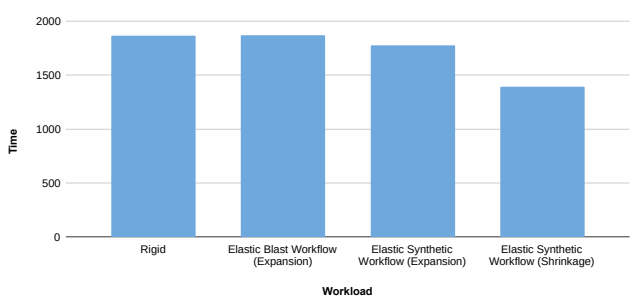
Malleable Scheduler

- Hierarchical scheduling mechanism with a simple custom FIFO scheduler written in Python with a file-based communication mechanism between the scheduler and workflows to exchange resource change events [4]







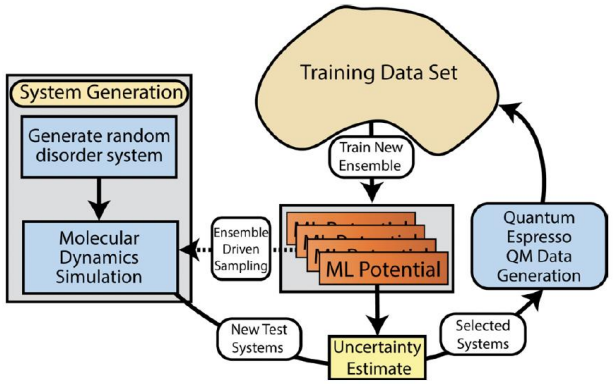


- Created workload with a combination of elastic and rigid jobs and ran in the elastic workflow system
- Blast and Synthetic Workflows and CloverLeaf [8] Application

Dynamic Resource Management can improve Resource Utilization.

Future Work and Summary

- Experiments with more real-world workflows Examol[5], and LANL's ALF [6]
- Advanced PMIx based malleable resource manager and integration with Slurm
- Utilize more PMIx functionalities



Active Learning Framework [6]

Summary:

- Dynamic resource management promise to improve the system performance in terms of execution time and resource utilization
- Proof of concept for use of PMIx as common standard for workflows, applications and resource managers to facilitate operations in HPC systems

Work in Progress...

Open to Collaborations!

Acknowledgements: Howard Pritchard (Los Alamos National Laboratory), Sheikh Ghafoor, (Tennessee Tech University)

References

1. Y. Babuji, A. Woodard, Z. Li, D. S. Katz, B. Clifford, R. Kumar, L. Lacinski, R. Chard, J. M. Wozniak, I. Foster, M. Wilde, and K. Chard, "Parsl: Pervasive parallel programming in python," in Proceedings of the 28th International Symposium on High-Performance Parallel and Distributed Computing, ser. HPDC '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 25–36. [Online]. Available: https://doi.org/10.1145/3307681.3325400
2. R. H. Castain, J. Hursey, A. Bouteiller, and D. Solt, "Pmix: Process management for exascale environments," Parallel Computing, vol. 79, pp. 9–29, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167819118302424
3. R. Bhattarai, H. Pritchard and S. Ghafoor, "Dynamic Resource Management for Elastic Scientific Workflows using PMIx," 2024 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), San Francisco, CA, USA, 2024, pp. 686-695, doi: 10.1109/IPDPSW63119.2024.00131.
4. R. Bhattarai, H. Pritchard and S. Ghafoor, " Evaluation of a Dynamic Resource Management Strategy for Elastic Scientific Workflows" Europar 2024 Workshop (Accepted)
5. https://github.com/exalearn/ExaMol
6. Smith, J.S., Nebgen, B., Mathew, N. *et al.* Automated discovery of a robust interatomic potential for aluminum. *Nat Commun* **12**, 1257 (2021). https://doi.org/10.1038/s41467-021-21376-0
7. A. Tarraf et al., "Malleability in Modern HPC Systems: Current Experiences, Challenges, and Future Opportunities," in IEEE Transactions on Parallel and Distributed Systems, vol. 35, no. 9, pp. 1551-1564, Sept. 2024, doi: 10.1109/TPDS.2024.3406764.
8. Mallinson, A.C., Beckingsale, D.A., Gaudin, W.P., Herdman, J.A., Levesque, J.M., & Jarvis, S.A. (2013). CloverLeaf: Preparing Hydrodynamics Codes for Exascale