I/O Characterization of Heterogeneous Workflows

Olga Kogiou*, Hariharan Devarajan+, Chen Wang+, Weikuan Yu*, Kathryn Mohror+

Florida State University*, Lawrence Livermore National Lab+

Overview and Contribution

Workflows:

- ❑ Are pre-defined or random ordered execution of a set of tasks that produce scientific results.
- ❑ Form Directed Acyclic Graphs (DAGs) where the nodes are the tasks and the edges are the inter data-dependencies.
- ❑ Consist of ***heterogeneous stages*** with different system requirements which spawn processes dynamically depending on their needs.

Project goals:

- ❑ To explore **Performance Optimization** opportunities for large-scale workflows.
- ❑ To improve **Resource Management** in High-Performance Computing (HPC) systems

Immediate goals:

- ❑ To investigate into the different requirements of heterogeneous workflows.
- ❑ To identify **Performance bottlenecks** in workflows.

Our **contributions** are:

- ❑ We analyze the run-time and perform a systematic **characterization** for three workflows.
- ❑ We identify stages that **have higher I/O bandwidth requirement,** are heterogeneous and can benefit from **Dynamic Resource Scheduling**.

Methodology

1. Workflow Source code annotation to trace Application calls
2. Calls are translated into `events` and stored into trace files of GB of size
3. Traces are loaded and analyzed using *dask distributed framework* and queries.

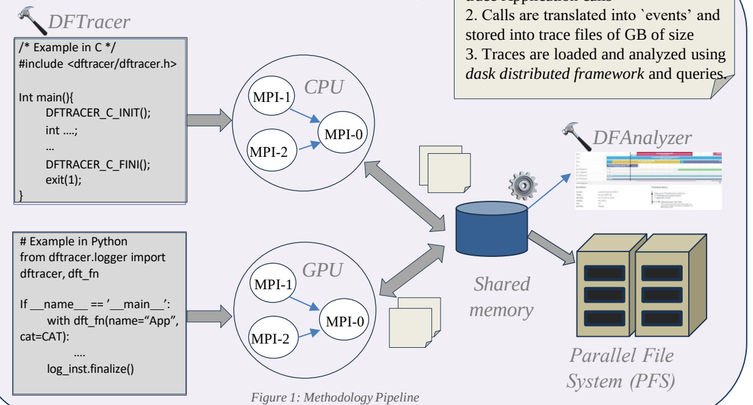


Figure 1: Methodology Pipeline

Workflow	LC Cluster	Nodes	Process per node	Input	How	Computing Element	PFS	Code	DFTracer Overhead
Montage [1]	Corona	32	48	2MASS survey	Flux, MPI	AMD Rome CPU	Lustre	C	7%
1000 Genome [2]	Corona	32	48	10 chromosomes from IGRS [4]	Flux, Pegasus Workflow Manager [5]	AMD Rome CPU	Lustre	Python	5%
MuMMI [3]	Lassen	32	44	PyTorch model file	Flux, Custom workflow manager	IBM Power9 CPU, NVIDIA V100 GPU	GPFS	C/C++ CUDA Python	<1%

Table I: Software, Hardware and Experimental Set-Up

Runtime Analysis Results

(a) Montage

Total Time	I/O Time	App Time
2hr 27 min	15 min	2hr 22min

Table II: Runtime Analysis of Montage

(b) 1000 Genome

Total Time	I/O Time	App Time
1hr 44min	48min 18sec	1hr 30min

Table III: Runtime Analysis of 1000 Genome

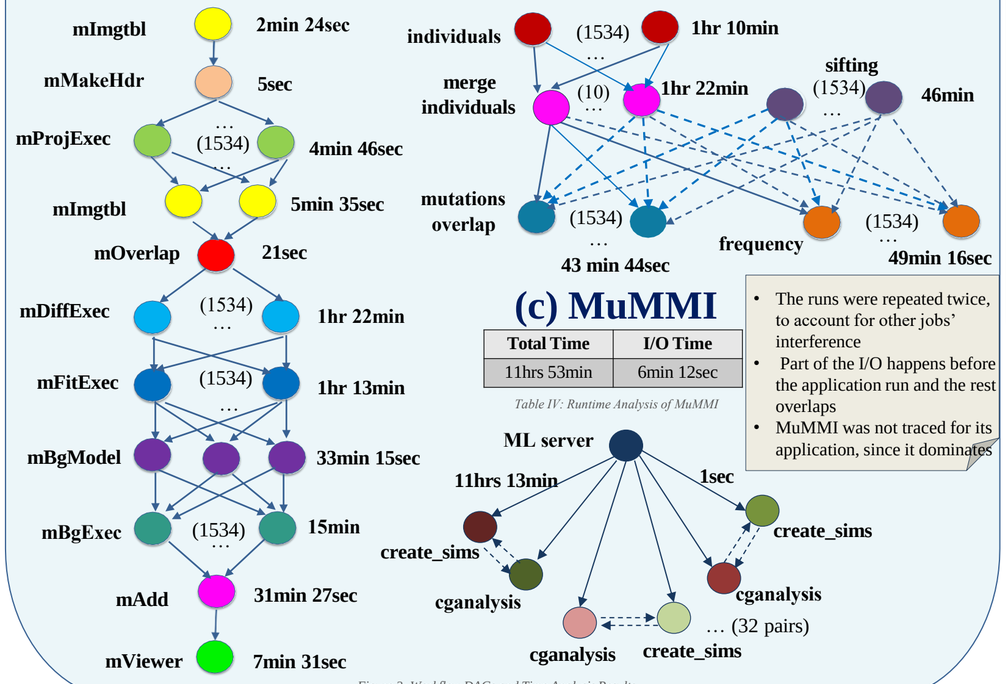


Figure 2: Workflow DAGs and Time Analysis Results

(c) MuMMI

Total Time	I/O Time
11hrs 53min	6min 12sec

Table IV: Runtime Analysis of MuMMI

- The runs were repeated twice, to account for other jobs' interference
- Part of the I/O happens before the application run and the rest overlaps
- MuMMI was not traced for its application, since it dominates

Scheduling and Bandwidth

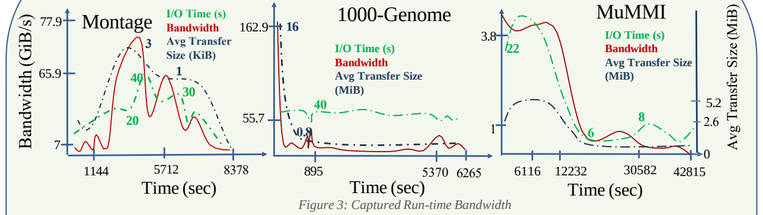
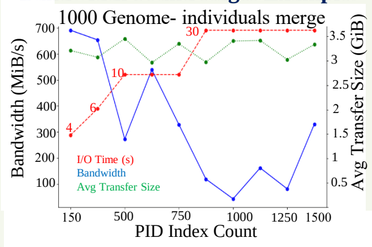


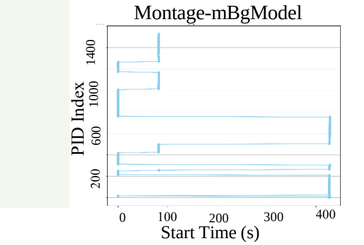
Figure 3: Captured Run-time Bandwidth

Bandwidth bound stages Example:



Cumulative bandwidth after progressively adding application instances plateaus.

Resource bound stages Example:



The Starting time of different PID indexes varies due to limited resources.

Figure 4: I/O bound stages limited by the Parallel File System parallelism or the resources

Findings

Montage:

- ❑ *mFitExec:* bandwidth-bound, Suggestion: data aggregation in the PFS itself.
- ❑ *mBgModel:* resource-bound, Suggestion: Dynamic Scheduling with Flux.

1000 Genome:

- ❑ *mutations overlap, merge individuals:* bandwidth-bound, Suggestion: use of isolated storage solutions such as node-local storage
- ❑ *individuals:* resource-bound , can be benefitted from Dynamic Scheduling.

MuMMI:

- ❑ It is not I/O bound
- ❑ Uses optimizations such as Flux hierarchical scheduling to isolate simulation-analysis pairs and shared memory usage.

I/O Characterization Results

Montage: mAdd (4.4%I/O) >> mImgtbl (3.22%I/O) >> mFitExec (2.2%I/O) >> mBgModel (1.5%I/O) >> others (<1%I/O)
1000 Genome: individuals merge (47%I/O) >> mutations overlap (45%I/O) >> frequency (43%I/O) >> individuals (9.9%I/O) >> sifting (1%I/O)
MuMMI: cganalysis (0.9%I/O) >> create_sims (<0.5%I/O)

Workflow	I/O thread	Application processes	Read size			I/O bound stage	Non I/O bound stage
			Max	Avg	Min		
Montage	42186	1534	1 MB	20 B	1 B	mFitExec	mBgExec
1000 Genome	2644	1534	16 MB	2 MB	5 KB	mutations overlap	sifting
MuMMI	22949	1408	588 MB	11 MB	6 KB	cganalysis	create_sims

Table V: Software, Hardware and Experimental Set-Up

References

1. J. C. Jacob, D. S. Katz, G. B. Berriman, J. C. Good, A. C. Laity, E. Deelman, C. Kesselman, G. Singh, M. H. Su, T. A. Prince, and R. Williams, "Montage: a grid portal and software toolkit for science-grade astronomical image mosaicking," International Journal of Computational Science and Engineering, vol. 4, no. 2, p. 73, 2009.
2. X. Zheng-Bradley et al., "Alignment of 1000 Genomes Project reads to reference assembly GRCh38," Gigascience, vol. 6, no. 7, pp. 1–8, 2017.
3. F. Di Natale, H. Bhatia, T. S. Carpenter, C. Neale, S. Kokkila- Schumacher, T. Oppelstrup, L. Stanton, X. Zhang, S. Sundram, T. R. W. Scogland, G. Dharuman, M. P. Surh, Y. Yang, C. Misale, L. Schneidenbach, C. Costa, C. Kim, B. D'Amora, S.Gnanakaran, D. V. Nissley, F. Streitz, F. C. Lightstone, P. T. Bremer, J. N. Glosli, and H. I. Ingo lfsson, "A massively parallel infrastructure for adaptive multiscale simulations: modeling RAS initiation pathway for cancer," in Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis. Denver Colorado: ACM, Nov. 2019, pp. 1–16.
4. https://www.internationalgenome.org
5. https://pegasus.isi.edu
6. https://github.com/hariharan-devarajan/dftracer
7. H. Devarajan, L. Pottier, K. Velusamy, H. Zheng, I. Yildirim, O. Kogiou, W. Yu, A. Kougkas, X.-H. Sun, J. S. Yeom, and K. Mohror, "DFTracer: An Analysis-Friendly Data Flow Tracer for AI-Driven Workflows," in SC24: International Conference for High Performance Computing, Networking, Storage and Analysis. Atlanta, GA: IEEE, Jun. 2024.

Acknowledgement

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344 (LLNL-POST-867997). This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research under the DOE Early Career Research Program. This work is supported in part by the National Science Foundation award 1763547, and has used the NoleLand facility that is funded by the U.S. National Science Foundation grant CNS-1822737. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.