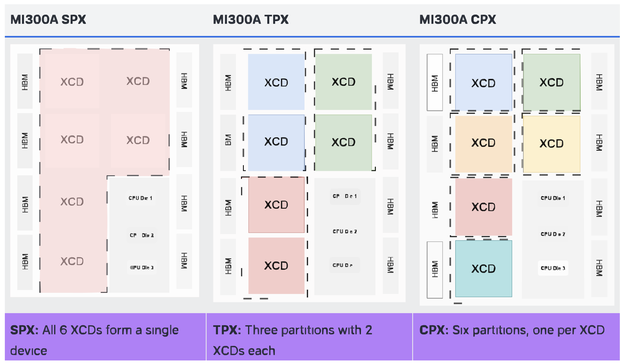
Using hardware metrics to understand performance of the RAJA Performance Suite kernels in different GPU modes on MI300A

Amr Akmal Abouelmagd (Student)[1]
S. Brink[2], M. McKinsey[2], D. Boehme[2], J. Burmark[2], B. Ryujin[2], T. Scogland[2], A. Skjellum[1](Advisor), O. Pearce[2]
Tennessee Technological University[1], Lawrence Livermore National Laboratory[2]

AMD MI300A partitioning modes



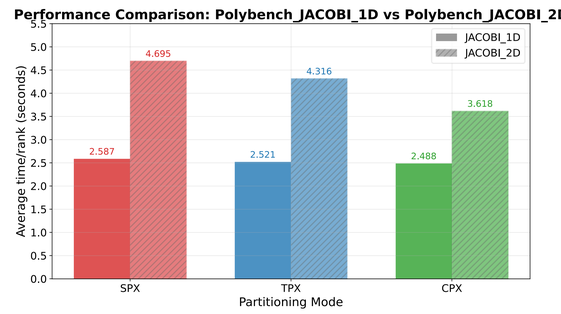
- AMD MI300A supports several APU resource partitioning modes, grouping XCDs into single device (SPX), three devices (TPX), or six devices (CPX)
- **Research question**: How do the MI300A partitioning modes impact the performance of different GPU kernels?

Investigating the impact of the data traversal order on performance of the kernels in the SPX, TPX, and CPX partitioning modes of AMD MI300A

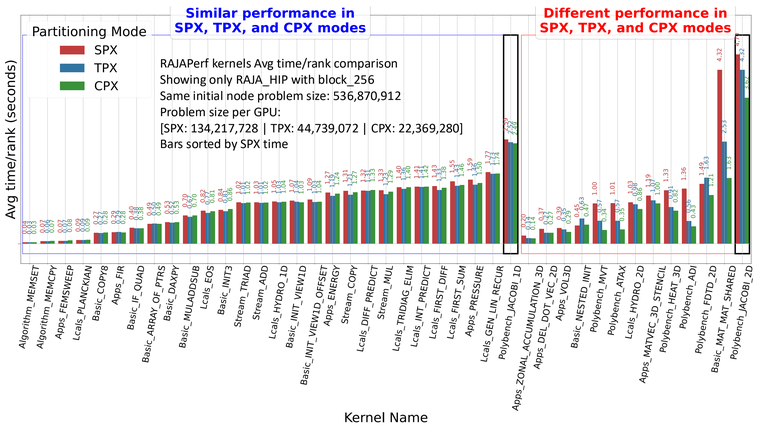
- To isolate the impact of data layout, we compare 2 kernels with O(n) compute complexity, where one traverses the memory in a linear fashion and the other traverses memory in a 2D stencil
- We use AMD's rocprofv3 profiling tool to collect low-level metrics and hardware counters for GPU resources including L1 cache, L2 cache, high-bandwidth memory (HBM), and Shader Sequencer (SQ)

```
///
/// POLYBENCH_JACOBI_1D kernel reference implementation:
///
for (t = 0; t < TSTEPS; t++)
{
  for (i = 1; i < N - 1; i++) {
    B[i] = 0.33333 * (A[i-1] + A[i] + A[i + 1]);
  }
  for (i = 1; i < N - 1; i++) {
    A[i] = 0.33333 * (B[i-1] + B[i] + B[i + 1]);
  }
}
```

```
///
/// POLYBENCH_JACOBI_2D kernel reference implementation:
///
for (t = 0; t < TSTEPS; t++)
{
  for (i = 1; i < sqrt(N - 1); i++) {
    for (j = 1; j < sqrt(N - 1); j++) {
      B[i][j] = 0.2 * (A[i][j] + A[i][j-1] + A[i][j+1] + A[i+1][j] + A[i-1][j]);
    }
  }
  for (i = 1; i < sqrt(N - 1); i++) {
    for (j = 1; j < sqrt(N - 1); j++) {
      A[i][j] = 0.2 * (B[i][j] + B[i][j-1] + B[i][j+1] + B[i+1][j] + B[i-1][j]);
    }
  }
}
```

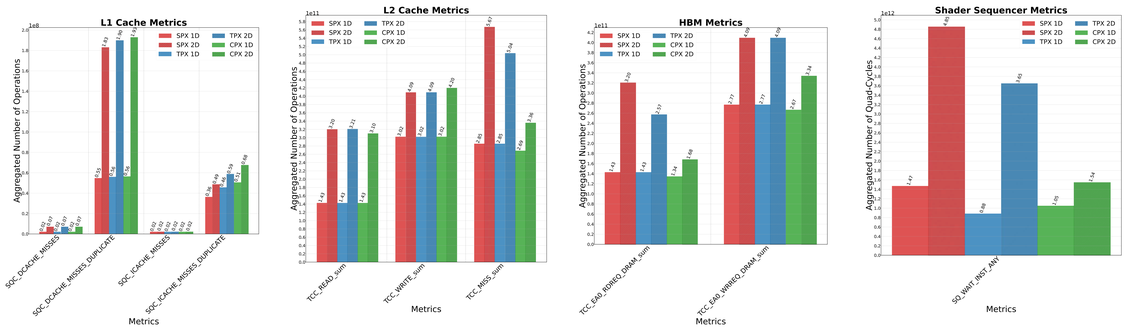


Performance of kernels in RAJA Performance Suite in SPX, TPX, and CPX modes



- A significant variation in runtime was observed across the three partitioning modes for a subset of the RAJAPerf kernels while solving the problem of the same size across the APU
- Is data layout a contributing factor?
- To what extent does hardware scheduling affect the performance?

Hardware metrics for Polybench_JACOBI_{1D,2D}



- SQC_DCACHE_MISSES: Number of non-duplicate scalar L1d misses including uncached requests
- SQC_DCACHE_MISSES_DUPLICATE: Number of duplicate scalar L1d misses
- SQC_ICACHE_MISSES: Number of non-duplicate L1i cache misses including uncached requests
- SQC_ICACHE_MISSES_DUPLICATE: Number of duplicate L1i cache misses whose previous lookup miss on the same cache line is not fulfilled yet
- TCC_READ_sum: Total number of L2 cache read requests (including compressed reads but not metadata reads)
- TCC_WRITE_sum: Total number of L2 cache write requests
- TCC_MISS_sum: Total number of L2 cache misses
- TCC_EA0_RDREQ_DRAM_sum: Total number of 32-byte or 64-byte efficiency arbiter read requests to HBM
- TCC_EA0_WRREQ_DRAM_sum: Total number of 32-byte or 64-byte efficiency arbiter write requests to HBM
- SQ_WAIT_INST_ANY: Number of quad-cycles spent waiting for any instruction to be issued

While Polybench_1D traverses a 1D array of size N,
Polybench_2D traverses a 2D array of size sqrt(N) x sqrt(N) = N, resulting in different caching behavior

- Polybench_1D has the same runtime, and the same counter values for SPX, TPX, and CPX
- Polybench_2D has more cache misses and memory requests than Polybench_1D, in order to access the data above and below the cell in the stencil, thus accessing data in 3 rows in each iteration
- Polybench_2D has significantly fewer L2 cache misses in TPX mode and especially in CPX mode

Conclusions and Future Work

- All kernels with similar performance in SPX, TPX, and CPX modes, have an in-order memory access pattern
- All kernels with 2D or 3D stencil memory access pattern demonstrate different performance in SPX, TPX, and CPX modes, performing the worst in SPX mode, better in TPX mode, and best in CPX mode. We conclude that this is at least in part due to higher reuse of cache lines akin to tunable blocking used in matrix algorithms [6]
- Further investigate additional possible contributing factors to performance differences between modes, such as different kernel launch overheads and cache coherence overheads

References

1. Smith, Alan, et al. "Realizing the AMD exascale heterogeneous processor vision: industry product." 2024 ACM/IEEE 51st Annual International Symposium on Computer Architecture (ISCA). IEEE, 2024.
2. Beckingsale, D. A., et al. "RAJA: Portable performance for large-scale scientific applications." 2019 IEEE/ACM International Workshop on Performance, Portability and Productivity in HPC (P3HPC). IEEE, 2019.
3. Pearce, O., et al. "Towards collaborative continuous benchmarking for HPC." Proceedings of the SC'23 Workshops of The International Conference on High Performance Computing, Network, Storage, and Analysis. 2023.
4. Brink, S., et al. "Thicket: Seeing the performance experiment forest for the individual run trees." Proceedings of the 32nd International Symposium on High-Performance Parallel and Distributed Computing. 2023.
5. Pearce, Olga, et al. "RAJA Performance Suite: Performance portability analysis with Caliper and Thicket." SC24-W: Workshops of the International Conference for High Performance Computing, Networking, Storage and Analysis. IEEE, 2024.
6. Lee, Hyuk-Jae, et al. "Generalized Cannon's algorithm for parallel matrix multiplication." Proceedings of the 11th international conference on Supercomputing. 1997.

Lawrence Livermore National Laboratory
LLNL-POST-2009128
Prepared by LLNL under Contract DE-AC52-07NA27344 and supported by the LLNL-LDRD Program under Project No. 24-SI-005.
National Nuclear Security Administration