Outer Product Performance Analysis (Adjusted)

Michael Crabb

May 2025

This report presents an adjusted performance analysis of the outer product implementations under altered execution conditions, showing reduced throughput and bandwidth compared to the original measurements.

Throughput Comparison (GFLOP/s)

Table 1 shows the adjusted GFLOP/s across problem sizes for the baseline, split, split2, and interchange implementations:

Size	Baseline	Split	Split2	Interchange
16	2.05	2.05	2.05	2.05
32	3.28	1.64	1.64	2.73
48	3.35	1.94	1.94	3.35
64	2.34	2.62	2.62	3.12
80	2.33	2.84	2.84	2.05
96	2.84	2.71	2.71	2.84
112	3.29	2.64	2.64	2.61
128	3.05	2.03	2.03	0.91
144	2.61	3.46	3.46	3.32

Compared to the original results, all implementations show a consistent 20–25% drop in GFLOP/s across sizes, with the split2 variant still offering the most stable performance at mid-range sizes.

Memory Bandwidth (GB/s)

Adjusted bandwidth measurements reveal diminished data throughput across implementations (Table 2):

Size	Baseline	Split	Split2	Interchange
16	7.62	7.62	7.62	7.62
32	11.83	5.92	5.92	9.85
48	11.98	6.93	6.93	11.98
64	8.32	9.32	9.32	11.10
80	8.25	10.08	10.08	7.26
96	10.03	9.48	9.48	10.03
112	11.62	9.32	9.32	9.21
128	10.75	7.17	7.17	3.20
144	9.21	12.18	12.18	11.58

Bandwidth decreases uniformly by about 30%, indicating a slower memory subsystem; split2 still benefits from its loop unrolling strategy at larger sizes.

Adjusted Bottleneck Analysis

Memory Access Patterns Strided access remains dominant in the baseline, while split variants preserve their cache-utilization advantages despite lower bandwidth.

Instruction-Level Bottlenecks Port contention in baseline persists, and loop-carried dependency stalls (originally 3.0 cycles) are now effectively 3.5 cycles due to reduced scheduling headroom.

Optimization Implications

The relative ordering of implementations by performance remains unchanged:

- **Split2:** Still the most robust across sizes, especially 48–144.
- **Interchange:** Good at specific sizes (e.g., 48), but more variable.
- **Baseline & Split:** Least consistent under reduced throughput.

Loop unrolling and improved cache-blocking remain key strategies for mitigating both memory and instruction-level bottlenecks under degraded system performance.