EML Task 11.2

1. Task 12.2.4 - Benchmarks

- Matrix size: 8192 (square)
- Iterations: 1
- Floating point operations: $2 * 8192^3 * \text{iterations} = 1.099.511.627.776$

Threads	Type	CPU time in s	GFLOPS
1	linalg.generic	423.914	2.594
1	linalg.matmul	39.648	27.734
1	TOSA	44.623	24.644
72	linalg.generic	520.927	2.110
72	linalg.matmul	40.488	27.157
72	TOSA	40.654	27.048

I find the results extremely surprising and I have no explanation as to why it turned out this way. Especially the "–task_topology_max_group_count=" option did not seem to work at all, but unfortunately I could not find any information on it online.

2. Task 12.2.5 - Microkernels

Microkernels are specialized, architecture-specific code snippets that are supposed to perform single core computational tasks efficiently.This means that they perform low-level, high-performance computations by making use of the features of different computer architectures.

As specified by the paper, they are compiled into self-contained bitcode files for each target architecture to ensure that the microkernels can be easily utilized across different platforms without relying on the standard library or OS-specific features. They also only take scalar parameters, including buffer pointers and strides. MLIR vector dialects are generally avoided.

-> What role do microkernels play?

Thanks to microkernels, we don't need to worry about performant implementations of general and commonly used computational tasks. The compiler will infer them automatically to ensure optimized performance on different architectures. The compiler is also able to inline those kernels which achieves "perfect results with no downsides compared to a pure code-generation approach."