CS 5220: Project 1 Initial Report

Group 018: Guantian Zheng (gz94), Stephen McDowell (sjm324)

1 Introduction

2 Permutations of Loop Variables

We experimented with different permutations of the three loop variables `i`, `j` and `k`. Apparently, `j, k, i` (starting from the outermost loop) gains the most advantage by reading in continuous blocks of matrix `C` and `A`. The loops are as follows:

```
for(j = 0; j < M; ++j) {
    for(k = 0; k < M; ++k) {
        double b_kj = B(k, j);
        for(i = 0; i < M; ++i) {
            C(i, j) += A(i, k) * b_kj;
        }
    }
}
```

3 Tuning the Block Size

Building on previous results, we decided to adopt the `j-k-i` loop for per-block computation (`basic_dgemm`), while searching for an appropriate block size.

Looks like reordered `blocked` with block size 256 takes the lead. Let's try with larger sizes:

The Gflops of `blocked_perm` keeps rising until size 2048, which means `j-k-i` looping blocked implementation with size 1024 is by far the optimal solution for our test cases.