Homework 3
All-Pairs Shortest Path Optimization

Parallel Programming 2020

Algorithm

- Johnson's algorithm
  - O(V^2 log(V) + E)
  - Fast for sparse graph
- Floyd-Warshall
  - O(V^3)
  - Fast for dense graph
- Blocked Floyd-Warshall

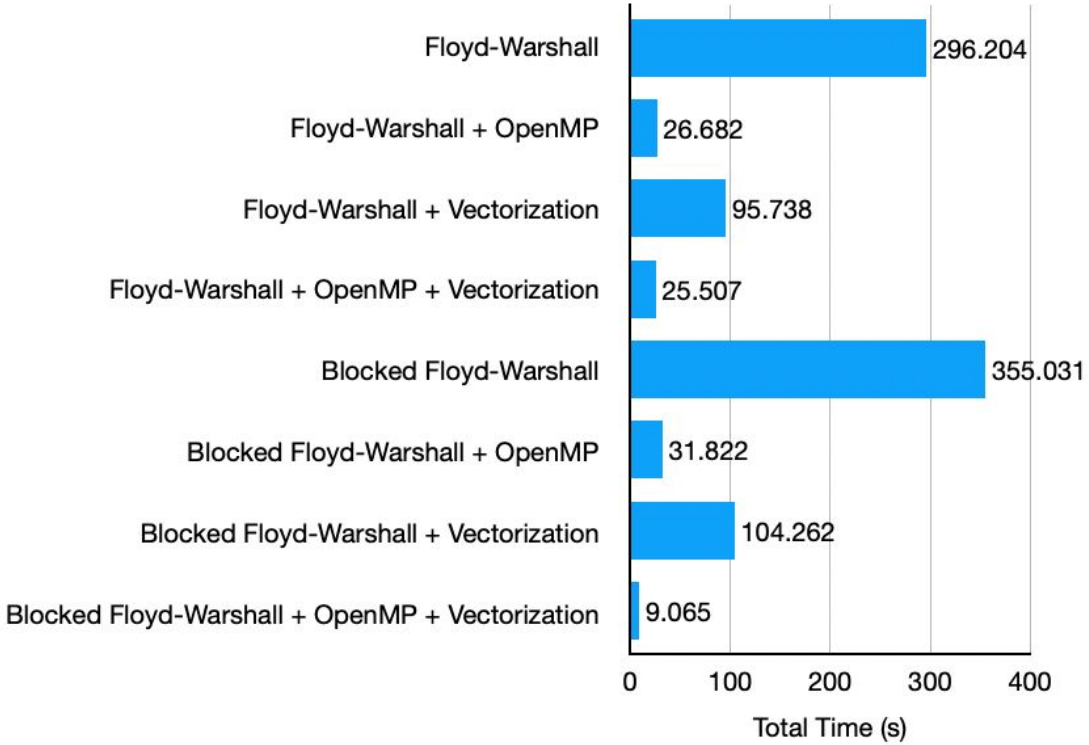
Parallelization

- Using OpenMP
- Static scheduler

Vectorization

```
__m128i as, bs, newdist, cs;
for (int k = 0; k < B; k++) {
   for (int i = 0; i < B; i++) {
       as  = _mm_set1_epi32(a[i * pvn + k]);
       for (int j = 0; j < B; j += 4) {
           bs = _mm_lddqu_si128((__m128i const*) &b[k * pvn + j]);
           newdist = _mm_add_epi32(as, bs);
           cs = _mm_lddqu_si128((__m128i const*) &c[i * pvn + j]);
           cs = _mm_min_epi32(cs, newdist);
           _mm_store_si128((__m128i*) &c[i * pvn + j], cs);
       }
   }
}
```

Blocked Floyd-Warshall



I/O

- mmap
- Read all data in one fread()

```
// read the distances to a buffer first
int* buffer = (int*)malloc(m * 3 * sizeof(int));
fread(buffer, sizeof(int), m * 3, file);
// write the distance from buffer to array
#pragma omp parallel for
for (int i = 0; i < m; i++) {
   Dist[buffer[i * 3]][buffer[i * 3 + 1]] = buffer[i * 3 + 2];
}
```