When we plan to submit our codes, we find the reason why OpenCL perform bad in our implementation. We set the local memory size to 1, which cause the bad performance.

// global, local size

const size\_t global[2] = {size, size + 1};

const size\_t local[2] = {1, 1};

So, here we run other size of local memory in OpenCL. We find that we get better performance. So we plan to update our term paper later.

The “5, 100, 500, 1000, 2000” is the size of matrix. The number in the table is the computation time. The unit is millisecond.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| GE  time(ms) | 5 | 100 | 500 | 1000 | 2000 |
| Serial | 0 | 0 | 50 | 1274 | 19112 |
| CUDA | 0 | 5 | 55 | 319 | 2231 |
| OpenCL(1,1) | 0 | 27 | 2282 | 17406 | 138856 |
| OpenCL(16,16) | 0 | 6 | 238 | 1328 | 16173 |
| OpenCL(32,32) | 0 | 7 | 346 | 1327 | 16182 |
| OpenCL(64,64) | 0 | 6 | 347 | 1325 | 9374 |
| OpenCL(128,128) | 0 | 7 | 239 | 17418 | 9385 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Gauss-Jordan time(ms) | 5 | 100 | 500 | 1000 | 2000 |
| Serial |  |  |  |  |  |
| CUDA | 0 | 3 | 41 | 239 | 1729 |
| OpenCL(1,1) | 0 | 17 | 1444 | 11351 | 90786 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Inversion  time(ms) | 5 | 100 | 500 | 1000 | 2000 |
| Serial | 10 | 90 | 1930 | 7990 |  |
| CUDA | 1 | 5 | 88 | 519 | 3888 |
| OpenCL(1, 1) | 0 | 21 | 1988 | 15799 | 125528 |