HEFFTE: Highly Efficient FFT for Exascale

1. Introduction

Considered one of the top 10 algorithms of the 20th century, the Fast Fourier Transform (FFT) is widely used by applications in science and engineering. Large scale parallel applications targeting exascale, such as those part of the Exascale Computing Project (ECP) in the USA, are designed for heterogeneous architectures and, currently, most of them rely on efficient state-of-the-art FFT libraries built as CPU kernels.

In this context, we have just released heFFTe [1,2] (pronounced hefft) library for FFT computations on heterogeneous platforms. We based our algorithm design on well-know libraries, FFTMPI [3] (used by EXAALT ECP-project) and SWFFT [4] (used by HACC project).

Using GPUs on multiple nodes, we achieve over 40x speedup on local kernels for 3D FFTs, and over 2x speedup for the whole computation. Fig. 1 shows heFFTe within the ECP software stack, and its dependences (e.g. cuBLAS, MAGMA).

2. Methodology and Algorithmic Design

In Fig. 2, we present the methodology called pencil-to-pencil, to compute a 3D FFT, this can be easily generalized for a general number of dimensions. It can be seen as a sequence 2 main tasks:

1. Computation of 1D FFTs: this can be done with standard 1DFFT libraries such as FFTW3, MKL, CUFFT, CLFFT, etc.
2. Tensor transposition or Reshape: (shown as the arrows) it involves 3 kernels, the first one for packing data in contiguous memory, a second one for inter-process communication, and the last one is the unpacking, performed at the receiving process.

The bottleneck for this algorithm is well-known to be the communication.

3. GPU speedup

Local CPU kernels presented on Section 2 are typical on state-of-the-art parallel FFT libraries, heFFTe provide new GPU kernels for these tasks achieving over 40x speedup.

<table>
<thead>
<tr>
<th>Packing</th>
<th>FFT computation</th>
<th>MPI communication</th>
</tr>
</thead>
<tbody>
<tr>
<td>9.65%</td>
<td>11.77%</td>
<td>49.43%</td>
</tr>
<tr>
<td>Unpacking</td>
<td>9.13%</td>
<td>6.72%</td>
</tr>
<tr>
<td></td>
<td>1.03%</td>
<td>97.34%</td>
</tr>
</tbody>
</table>

Local kernels accelerated 43x using GPUs.

Total speedup 2x

4. Scalability

The GPU version of heFFTe has very good weak and strong scalability, and achieves around 20 gigaFlops/s, which is over 2x the performance reported by state-of-the-art libraries FFTMPI [3] and SWFFT [4] for these sizes. We use 24 MPIs/node on each case, 1MPI/core for heFFTe CPU and 4MPI/GPU-Volta100 for heFFTe GPU.

5. Communication Bottleneck

Given these issues, we have developed a routine called heffte_alltoall which provides several options (that can be tuned) to perform an all-to-all communication, taking advantage of non-blocking MPI routines.

6. References

[1] https://bitbucket.org/ld/heffte/