CuTensor-Tubal: Optimized GPU Library for Low-Tubal-Rank Tensors

Tao Zhang[1,3], Xiao-Yang Liu[2] (co-primary author)

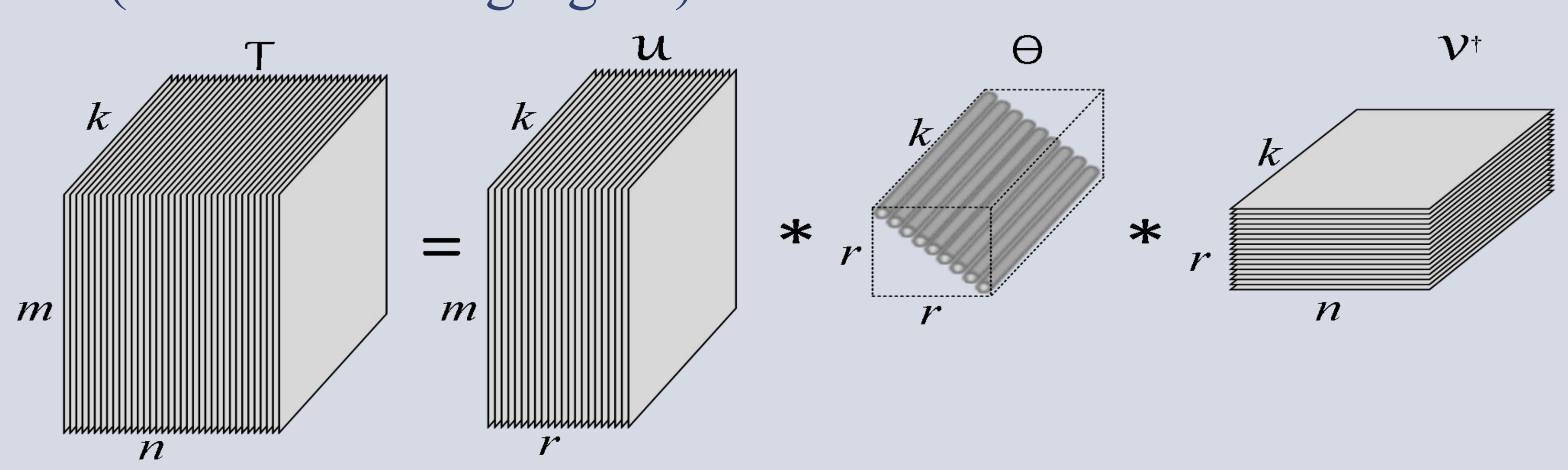
[1]Shanghai University [2] Columbia University [3] Shanghai Institute for Advanced Communication and Data Science

Introduction

- **Motivation**: real data are often modeled as low-rank tensors. However, tensor operations are compute-intensive. The running time and complexity grow rapidly with tensor order and size.
- **Our method**: design, implement and optimize a set of common tensor operations on GPUs based on the the transform-based low-tubal-rank tensor model.
- **Key challenges**: parallelization schemes, data transfer, memory access, hardware utilization.
- **Result**: a high-performance "cuTensor-tubal" GPU library with four tensor operations: t-FFT, inverse t-FFT, t-product, t-SVD.

Low-tubal-rank Tensor Model

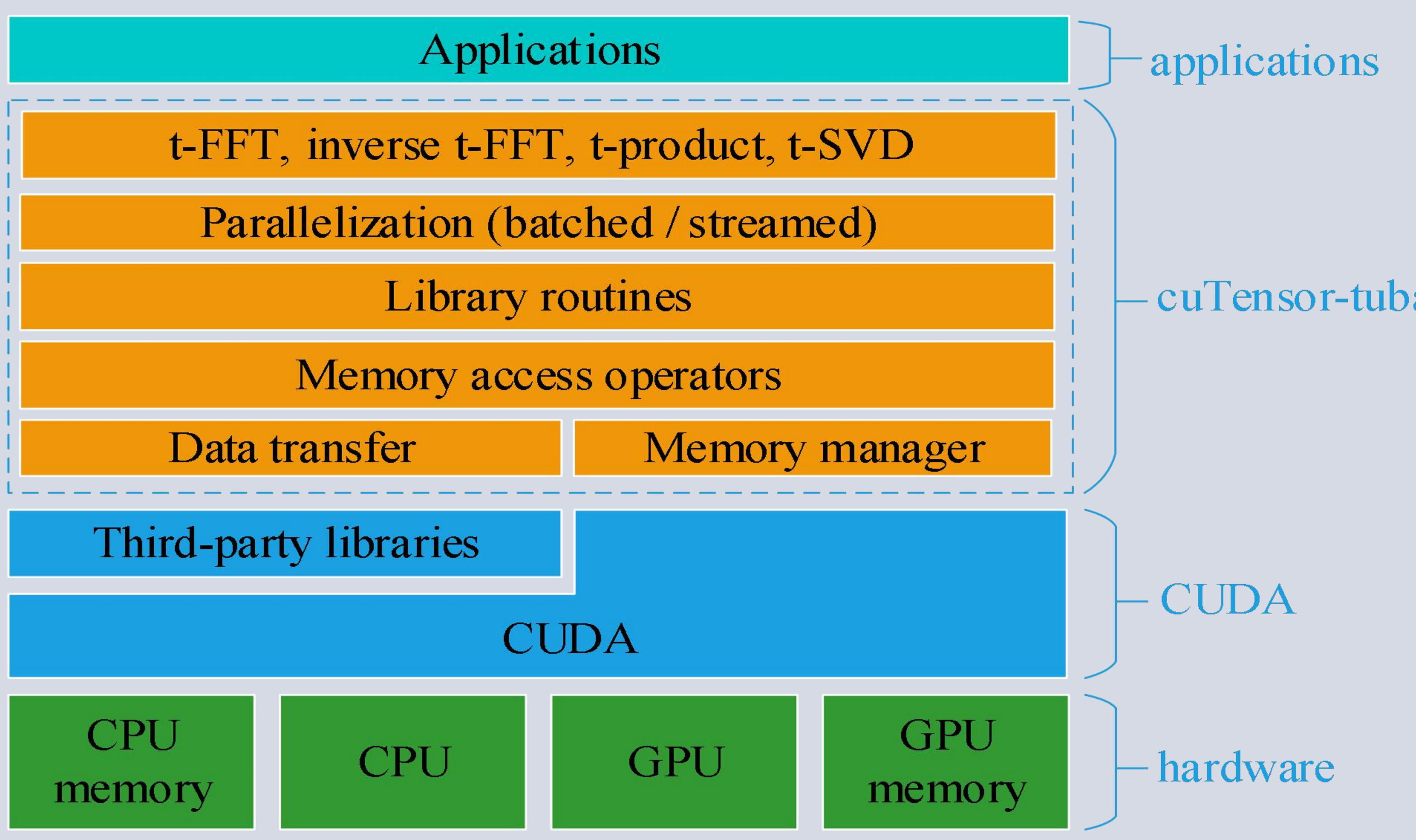
The low-tubal-rank tensor model is defined on the tensor Singular Value Decomposition (t-SVD). The model defines a set of tensor operations including tensor transpose, FFT, product, and SVD (as the following figure).



The cuTensor-tubal Library

1. Overview of the library

The cuTensor-tubal library is designed based on the GPU hardware, CUDA, and third-party libraries such as MAGMA.



2. Compute tensor operations in the frequency domain

1) converting the input tensor into the frequency domain by performing Fourier transform along the third-dimension (tube-wise DFT), called the t-FFT;

2) performing multiple independent (complex) matrix operations that possess strong parallelism;

3) converting the frequency domain results back to the time domain, called the inverse t-FFT.

3. Efficient data transfer

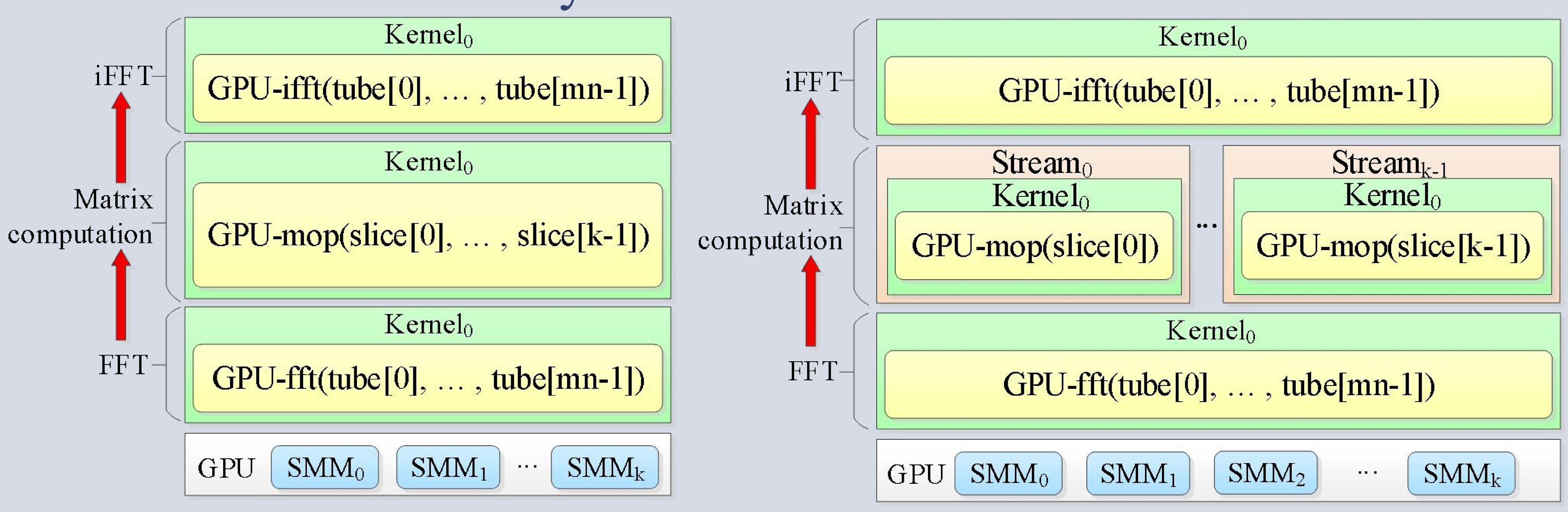
Overlap CPU to GPU data transfer with Fourier transforms for multi-input tensor operations sch as t-product. Overlap GPU to CPU data transfer with inverse Fourier transforms for multi-output tensor operations such as t-SVD.

4. Uniform memory access

cuTenor-tubal designs four memory access operators: tube-strided-fetch, tube-strided-store, slice-fetch, slice-store.

5. Two parallelization schemes

The batched scheme for synchronous execution versus the streamed scheme for asynchronous execution.



(a) Batched scheme. (b) Streamed scheme.

6. Exploit conjugate symmetry to save computation

Compute for half the slices and get the rest slices by:

$$\widetilde{\mathcal{A}}^{(\ell)} = \text{conjugate}(\widetilde{\mathcal{A}}^{(k-\ell+2)}), \quad \ell = \lceil \tfrac{k+1}{2} \rceil + 1, ..., k$$

Experiment Results

We evaluate the performance of cuTensor-tubal on a Tesla V100 GPU versus dual Intel Xeon E5-2640 V4 CPUs. The GPU t-product and t-SVD achieve up to 16.91X and 27.03X speedups versus that runs on two CPUs, respectively.

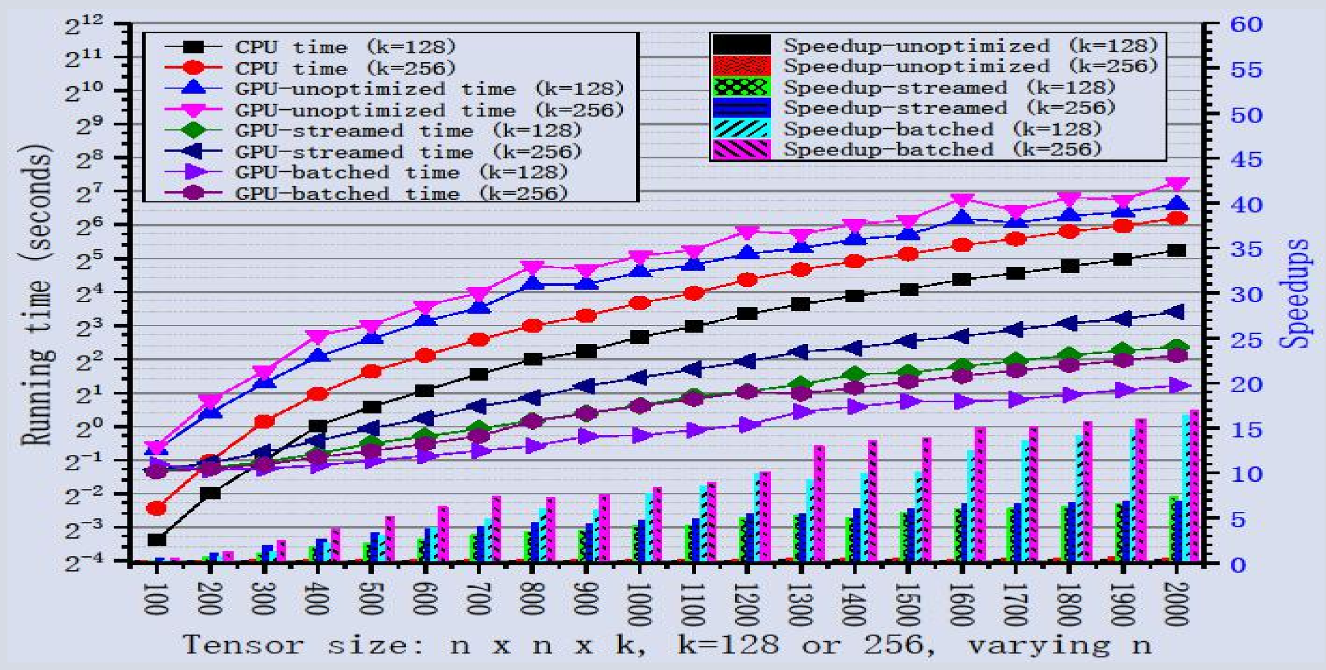


Fig. 1 Running time and speedups of t-product.

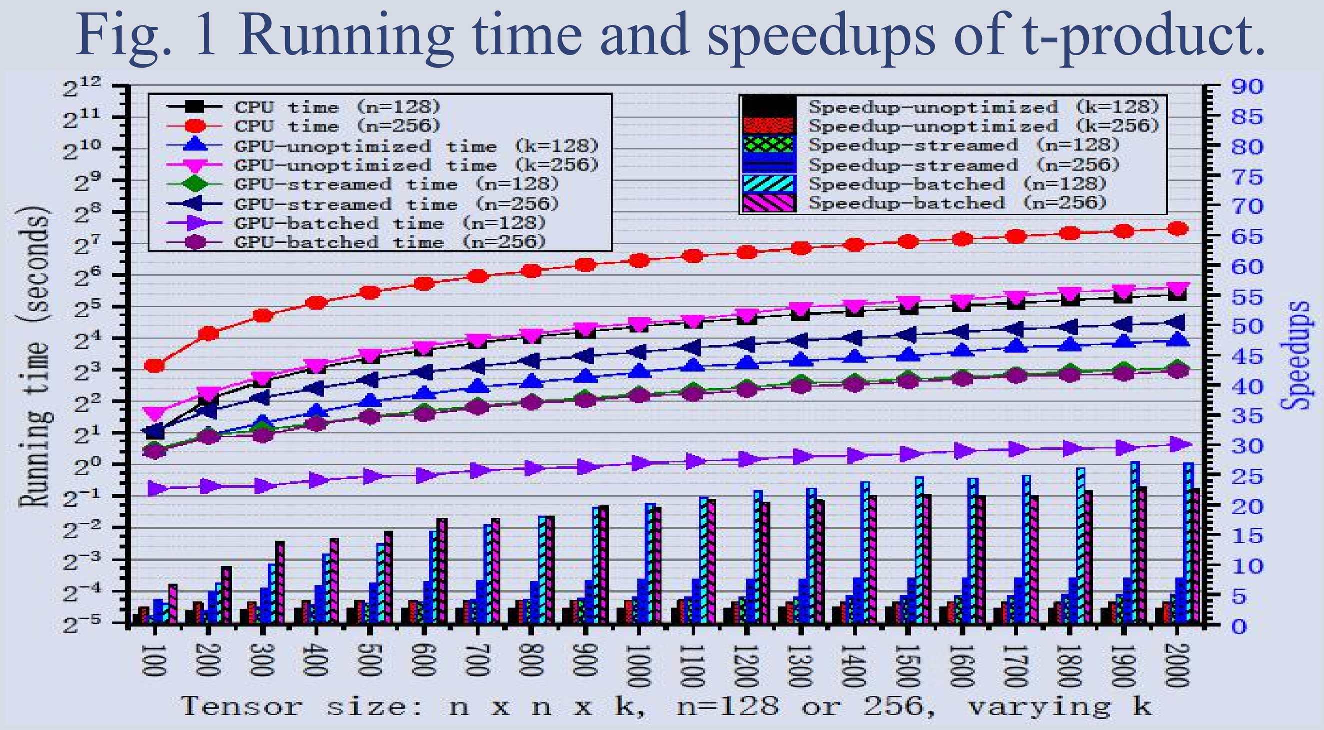


Fig. 2 Running time and speedups of t-SVD.

cuTensor-tubal has been extended to support more tensor operations including: t-QR, t-inverse, t-normalization.

Code available at: www.tensorlet.com

计算机工程与科学学院 School of Computer Engineering and Science Shanghai University

COLUMBIA | ENGINEERING The Fu Foundation School of Engineering and Applied Science